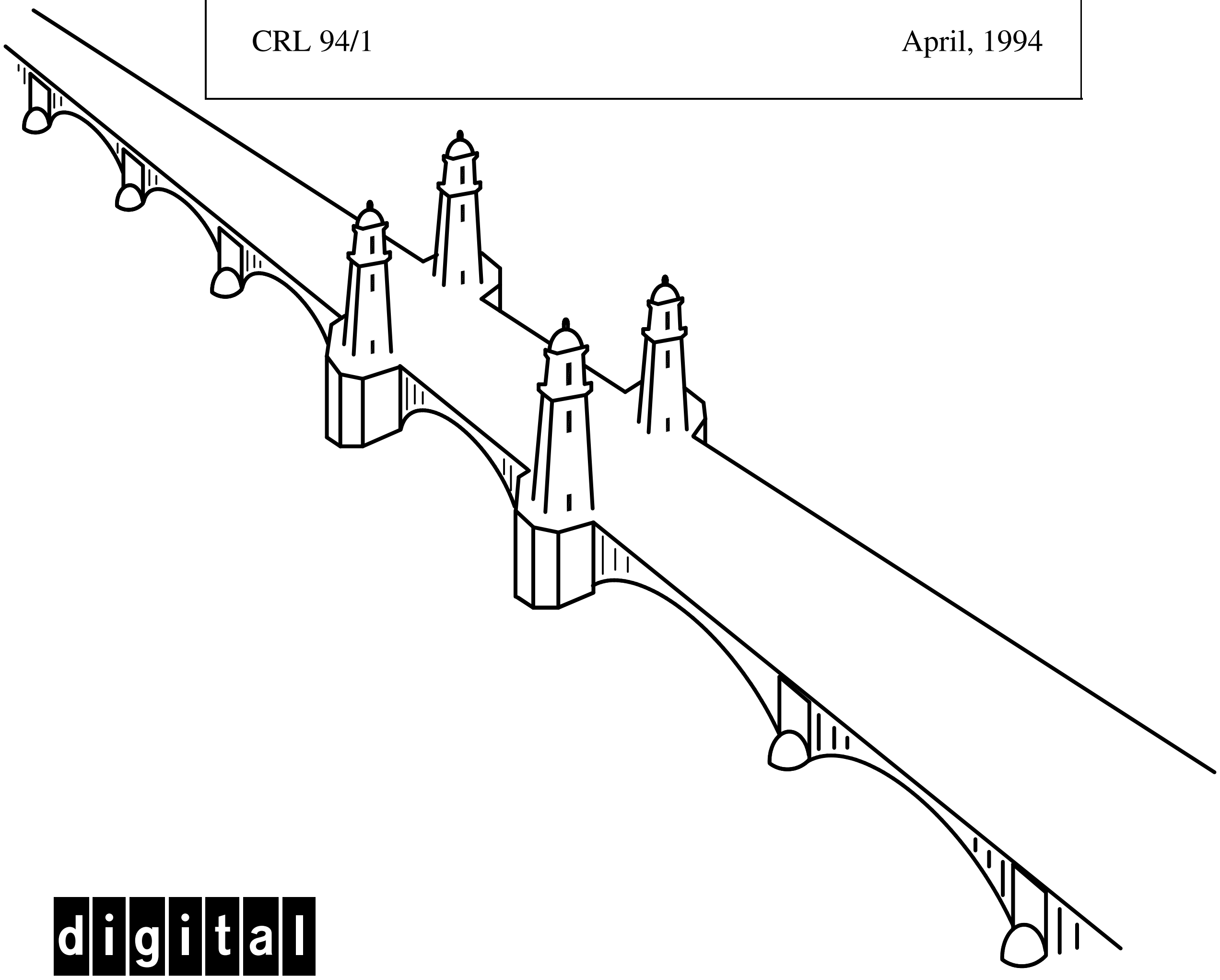
# Spline-Based Image Registration

Richard Szeliski and James Coughlan

Digital Equipment Corporation
Cambridge Research Lab

CRL 94/1 April, 1994



CAMBRIDGE RESEARCH LABORATORY
Technical Report Series

Digital Equipment Corporation has four research facilities: the Systems Research Center and the Western Research Laboratory, both in Palo Alto, California; the Paris Research Laboratory, in Paris; and the Cambridge Research Laboratory, in Cambridge, Massachusetts.

The Cambridge laboratory became operational in 1988 and is located at One Kendall Square, near MIT. CRL engages in computing research to extend the state of the computing art in areas likely to be important to Digital and its customers in future years. CRL's main focus is applications technology; that is, the creation of knowledge and tools useful for the preparation of important classes of applications.

CRL Technical Reports can be ordered by electronic mail. To receive instructions, send a message to one of the following addresses, with the word **help** in the Subject line:

| | |
|---|---|
| On Digital's EASYnet: | CRL::TECHREPORTS |
| On the Internet: | techreports@crl.dec.com |



Cambridge Research Laboratory
One Kendall Square
Cambridge, Massachusetts 02139

# Spline-Based Image Registration

Richard Szeliski and James Coughlan[1]

Digital Equipment Corporation
Cambridge Research Lab

CRL 94/1 April, 1994

**Abstract**

The problem of image registration subsumes a number of problems and techniques in multiframe image analysis, including the computation of optic flow (general pixel-based motion), stereo correspondence, structure from motion, and feature tracking. We present a new registration algorithm based on *spline* representations of the displacement field which can be specialized to solve all of the above mentioned problems. In particular, we show how to compute local flow, global (parametric) flow, rigid flow resulting from camera egomotion, and multiframe versions of the above problems. Using a spline-based description of the flow removes the need for overlapping correlation windows, and produces an explicit measure of the correlation between adjacent flow estimates. We demonstrate our algorithm on multiframe image registration and the recovery of 3D projective scene geometry. We also provide results on a number of standard motion sequences.

**Keywords:** motion analysis, multiframe image analysis, hierarchical image registration, optical flow, splines, global motion models, structure from motion, direct motion estimation.



[1]Harvard University, Department of Physics, Cambridge, MA 02138

# Contents

**1 Introduction** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **1**

**2 Previous work** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **2**

**3 General problem formulation** . . . . . . . . . . . . . . . . . . . . . . . . . . . . **4**

3.1 Spline-based motion model . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

**4 Local (general) flow estimation** . . . . . . . . . . . . . . . . . . . . . . . . . . . **9**

**5 Global (planar) flow estimation** . . . . . . . . . . . . . . . . . . . . . . . . . . **12**

**6 Mixed global and local (rigid) flow estimation** . . . . . . . . . . . . . . . . . . . . **18**

**7 Multiframe flow estimation** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **22**

**8 Experimental results** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **23**

**9 Discussion** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . **30**

**10 Future work and Conclusions** . . . . . . . . . . . . . . . . . . . . . . . . . . . . **32**

# List of Figures

1 Displacement spline: the spline control vertices $\{(\hat{u}_j, \hat{v}_j)\}$ are shown as circles ($\circ$) and the pixel displacements $\{(u_i, v_i)\}$ are shown as pluses ($+$). . . . . . . . . . . 6
2 Spline basis functions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
3 Block diagram of spline-based image registration . . . . . . . . . . . . . . . . . 13
4 Example of general flow computation . . . . . . . . . . . . . . . . . . . . . . . 14
5 Block diagram of global motion estimation . . . . . . . . . . . . . . . . . . . . 16
6 Example of affine flow computation: translation . . . . . . . . . . . . . . . . . . 17
7 Example of affine flow computation: divergence (zoom) . . . . . . . . . . . . . . 17
8 Example of 2D projective motion estimation . . . . . . . . . . . . . . . . . . . . 19
9 Example of 3D projective (rigid) motion estimation . . . . . . . . . . . . . . . . 21
10 **Sinusoid 1** and **Square 2** sample images . . . . . . . . . . . . . . . . . . . 25
11 **Yosemite** sample image and flow (unthresholded) . . . . . . . . . . . . . . . . . 28
12 **NASA Sequence**, rigid flow . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
13 **Rubik Cube** sequence, sparse flow . . . . . . . . . . . . . . . . . . . . . . . . 29
14 **Hamburg Taxi** sequence, dense local flow . . . . . . . . . . . . . . . . . . . . 30

# List of Tables

1 Summary of **Sinusoid 1** results . . . . . . . . . . . . . . . . . . . . . . . . . 24
2 Summary of **Square 2** results . . . . . . . . . . . . . . . . . . . . . . . . . . 26
3 Summary of **Translating Tree** results . . . . . . . . . . . . . . . . . . . . . . 26
4 Summary of **Diverging Tree** results . . . . . . . . . . . . . . . . . . . . . . . 27
5 Summary of **Yosemite** results . . . . . . . . . . . . . . . . . . . . . . . . . . 27

# 1 Introduction

The analysis of image sequences (*motion analysis*) is one of the more actively studied areas of computer vision and image processing. The estimation of motion has many diverse applications, including video compression, the extraction of 3D scene geometry and camera motion, robot navigation, and the registration of multiple images. The common problem is to determine correspondences between various parts of images in a sequence. This problem is often called *motion estimation*, *multiple view analysis*, or *image registration*.

Motion analysis subsumes a number of sub-problems and associated solution techniques, including optic flow, stereo and multiframe stereo, egomotion estimation, and feature detection and tracking. Each of these approaches makes different assumptions about the nature of the scene and the results to be computed (computational theory and representation) and the techniques used to compute these results (algorithm).

In this paper, we present a general motion estimation framework which can be specialized to solve a number of these sub-problems. Like Bergen *et al.* [1992], we view motion estimation as an image registration task with a fixed computational theory (optimality criterion), and view each sub-problem as an instantiation of a particular global or local motion model. For example, the motion may be completely general, it can depend on a few global parameters (e.g., affine flow), or it can result from the rigid motion of a 3D scene. We also use coarse-to-fine (*hierarchical*) algorithms to handle large displacements.

The key difference between our framework and previous algorithms is that we represent the local motion field using multi-resolution splines. This has a number of advantages over previous approaches. The splines impose an implicit smoothness on the motion field, removing in many instances the need for additional smoothness constraints (regularization). The splines also remove the need for correlation windows centered at each pixel, which are computationally expensive and implicitly assume a local translational model. Furthermore, they provide an explicit measure of the correlation between adjacent motion estimates.

The algorithm we develop to estimate structure and motion from rigid scenes differs from previous algorithms by using a general camera model, which eliminates the need to know the intrinsic camera calibration parameters. This results in estimates of *projective depth* rather than true Euclidean depth; the conversion to Euclidean shape, if required, can be performed in a later post-processing stage [Szeliski, 1994b].

The remainder of the paper is structured as follows. Section 2 presents a review of relevant previous work. Section 3 gives the general problem formulations for image registration. Section 4 develops our algorithm for local motion estimation. Section 5 presents the algorithm for global (planar) motion estimation. Section 6 presents our novel formulation of structure from motion based on the recovery of projective depth. Section 7 generalizes our previous algorithms to multiple frames and examines the resulting performance improvements. Section 8 presents experimental results based on some commonly used motion test sequences. Finally, we close with a comparison of our approach to previous algorithms and a discussion of future work.

# 2 Previous work

A large number of motion estimation and image registration algorithms have been developed in the past [Brown, 1992]. These algorithms include optical flow (general motion) estimators, global parametric motion estimators, constrained motion estimators (*direct methods*), stereo and multi-frame stereo, hierarchical (coarse-to-fine) methods, feature trackers, and feature-based registration techniques. We will use this rough taxonomy to briefly review previous work, while recognizing that these algorithms overlap and that many algorithms use ideas from several of these categories.

The general motion estimation problem is often called *optical flow* recovery [Horn and Schunck, 1981]. This involves estimating an independent displacement vector for each pixel in an image. Approaches to this problem include gradient-based approaches based on the *brightness constraint* [Horn and Schunck, 1981; Lucas and Kanade, 1981; Nagel, 1987], correlation-based techniques such as the Sum of Squared Differences (SSD) [Anandan, 1989], spatio-temporal filtering [Adelson and Bergen, 1985; Heeger, 1987; Fleet and Jepson, 1990], and regularization [Horn and Schunck, 1981; Hildreth, 1986; Poggio *et al.*, 1985]. Nagel [1987] and Anandan [Anandan, 1989] provide comparisons and derive relations between different techniques, while Barron *et al.* [1994] provide some numerical comparisons.

Global motion estimators [Lucas, 1984; Bergen *et al.*, 1992] use a simple flow field model parameterized by a small number of unknown variables. Examples of global motion models include affine and quadratic flow fields. In the taxonomy of Bergen *et al.* [1992], these fields are called parametric motion models, since they can be used locally as well (e.g., you can estimate affine flow at every pixel from filter outputs [Manmatha and Oliensis, 1992]).[1] Global methods are most useful when the scene has a particularly simple form, e.g., when the scene is planar.

Constrained (*quasi-parametric* [Bergen *et al.*, 1992]) motion models fall between local and global methods. Typically, these use a combination of global egomotion parameters with local shape (depth) parameters. Examples of this approach include the *direct methods* of Horn and Weldon [1988] and others [Hanna, 1991; Bergen *et al.*, 1992]. In this paper, we use projective descriptions of motion and depth [Faugeras, 1992; Mohr *et al.*, 1993; Szeliski and Kang, 1994] for our constrained motion model, which removes the need for calibrated cameras.

Stereo matching [Barnard and Fischler, 1982; Quam, 1984; Dhond and Aggarwal, 1989] is traditionally considered as a separate sub-discipline within computer vision (and, of course, photogrammetry), but there are strong connections between the two problems. Stereo can be viewed as a simplified version of the constrained motion model where the egomotion parameters (the *epipolar geometry*) are given, so that each flow vector is constrained to lie along a known line. While stereo is traditionally performed on pairs of images, more recent algorithms use sequences of images (*multiframe stereo* or *motion stereo*) [Bolles *et al.*, 1987; Matthies *et al.*, 1989; Okutomi and Kanade, 1992; Okutomi and Kanade, 1993].

Hierarchical (coarse-to-fine) matching algorithms have a long history of use both in stereo matching [Quam, 1984; Witkin *et al.*, 1987] and in motion estimation [Enkelmann, 1988; Anandan, 1989; Singh, 1990; Bergen *et al.*, 1992]. Hierarchical algorithms first solve the matching problem on smaller, lower-resolution images and then use these to initialize higher-resolution estimates. Their advantages include both increased computation efficiency and the ability to find better solutions (escape from local minima).

Tracking individual features (corners, points, lines) in images has always been alternative to iconic (pixel-based) optic flow techniques [Dreschler and Nagel, 1982; Sethi and Jain, 1987; Zheng and Chellappa, 1992]. This has the advantage of requiring less computation and of being less sensitive to lighting variation. The algorithm presented in this paper is closely related to patch-based feature trackers [Lucas and Kanade, 1981; Rehg and Witkin, 1991; Tomasi and Kanade, 1992]. In fact, our general motion estimator can be used as a parallel, adaptive feature tracker by selecting spline control vertices with low uncertainty in both motion components. Like [Rehg and Witkin, 1991], which is an affine-patch based tracker, it can handle large deformations in the

[1]The spline-based flow fields we describe in the next section can be viewed as local parametric models, since the flow within each spline patch is defined by a small number of control vertices.

patches being tracked.

Spline-based image registration techniques have been used in both the image processing and computer graphics communities. The work in [Goshtasby, 1986; Goshtasby, 1988] applies surface fitting to discrete displacement estimates based on feature correspondences to obtain a smooth displacement field. Wolberg [1990] provides a review of the extensive literature in digital image warping, which can be used to resample images once the (usually global) displacements are known. Spline-based displacement fields have recently been used in computer graphics to perform *morphing* [Beier and Neely, 1992] (deformable image blending) using manually specified correspondences. Registration techniques based on elastic deformations of images [Burr, 1981; Bajcsy and Broit, 1982; Bajcsy and Kovacic, 1989; Amit, 1993] also sometimes use splines as their representation [Bajcsy and Broit, 1982].

# 3 General problem formulation

The general image registration problem can be formulated as follows. We are given a sequence of images $I_t(x, y)$ which we assume were formed by locally displacing a reference image $I(x, y)$ with horizontal and vertical displacement fields[2] $u_t(x, y)$ and $v_t(x, y)$, i.e.,

$$I_t(x + u_t, y + v_t) = I(x, y). \tag{1}$$

Each individual image is assumed to be corrupted with uniform white Gaussian noise. We also ignore possible occlusions ("foldovers") in the warped images.

Given such a sequence of images, we wish to simultaneously recover the displacement fields $u_t$ and $v_t$ and the reference image $I(x, y)$. The maximum likelihood solution to this problem is well known [Szeliski, 1989], and consists of minimizing the squared error

$$\sum_t \int \int [I_t(x + u_t, y + v_t) - I(x, y)]^2 dx\, dy. \tag{2}$$

In practice, we are usually given a set of discretely sampled images, so we replace the above integrals with summations over the set of pixels $\{(x_i, y_i)\}$.

If the displacement fields $u_t$ and $v_t$ at different times are independent of each other and the reference intensity image $I(x, y)$ is assumed to be known, the above minimization problem

[2] We will use the terms *displacement field*, *flow field*, and *motion estimate* interchangeably.

decomposes into a set of independent minimizations, one for each frame. For now, we will assume that this is the case, and only study the two frame problem, which can be rewritten as[3]

$$E(\{u_i, v_i\}) = \sum_i [I_1(x_i + u_i, y_i + v_i) - I_0(x_i, y_i)]^2. \tag{3}$$

This equation is called the *Sum of Squared Differences* (SSD) formula [Anandan, 1989]. Expanding $I_1$ in a first order Taylor series expansion in $(u_i, v_i)$ yields the the *image brightness constraint* [Horn and Schunck, 1981; Anandan, 1989].

The above minimization problem will typically have many locally optimal solutions (in terms of the $\{(u_i, v_i)\}$). The choice of method for finding the best estimate efficiently is what typically differentiates between various motion estimation algorithms. For example, the SSD algorithm performs the summation at each pixel over a $5 \times 5$ window [Anandan, 1989] (more recent variations use adaptive windows [Okutomi and Kanade, 1992] and multiple frames [Okutomi and Kanade, 1993]). Regularization-based algorithms add smoothness constraints on the $u$ and $v$ fields to obtain good solutions [Horn and Schunck, 1981; Hildreth, 1986; Poggio *et al.*, 1985]. Multiscale or hierarchical (coarse-to-fine) techniques are often used to speed the search for the optimum displacement field.

Another decision that must be made is how to represent the $(u, v)$ fields. Assigning an independent estimate at each pixel $(u_i, v_i)$ is the most commonly made choice, but global motion descriptors are also possible [Lucas, 1984; Bergen *et al.*, 1992] (see also Section 5). Constrained motion models which combine a global rigid motion description with a local depth estimate are also used [Horn and Weldon Jr., 1988; Hanna, 1991; Bergen *et al.*, 1992], and we will study these in Section 6.

Both local correlation windows (as in SSD) and global smoothness constraints attempt to disambiguate possible motion field estimates by aggregating information from neighboring pixels. The resulting displacement estimates are therefore highly correlated. While it is possible to analyze the correlations induced by overlapping windows [Matthies *et al.*, 1989] and regularization [Szeliski, 1989], the procedures are cumbersome and rarely used.

## 3.1 Spline-based motion model

[3] We will return to the problem of multiframe motion in Section 7.

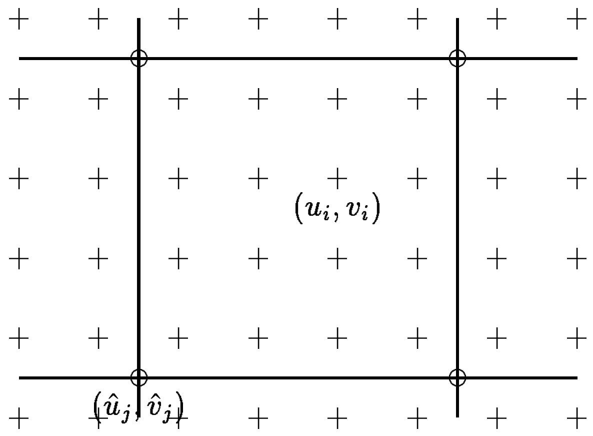



Figure 1: Displacement spline: the spline control vertices $\{(\hat{u}_j, \hat{v}_j)\}$ are shown as circles ($\circ$) and the pixel displacements $\{(u_i, v_i)\}$ are shown as pluses ($+$).

The alternative to these approaches, which we introduce in this paper, is to represent the displacements fields $u(x, y)$ and $v(x, y)$ as two-dimensional *splines* controlled by a smaller number of displacement estimates $\hat{u}_j$ and $\hat{v}_j$ which lie on a coarser *spline control grid* (Figure 1). The value for the displacement at a pixel $i$ can be written as

$$u(x_i, y_i) = \sum_j \hat{u}_j B_j(x_i, y_i) \quad \text{or} \quad u_i = \sum_j \hat{u}_j w_{ij}, \tag{4}$$

where the $B_j(x, y)$ are called the *basis functions* and are only non-zero over a small interval (*finite support*). We call the $w_{ij} = B_j(x_i, y_i)$ *weights* to emphasize that the $(u_i, v_i)$ are known linear combinations of the $(\hat{u}_j, \hat{v}_j)$.[4]

In our current implementation, the basis functions are spatially shifted versions of each other, i.e., $B_j(x, y) = B(x - \hat{x}_j, y - \hat{y}_j)$. We have studied five different interpolation functions:

1. block: $B(x, y) = 1$ on $[0, 1]^2$
2. linear: $B(x, y) = \begin{cases} (1 - x - y) & \text{on} \quad [0, 1]^2, \\ (x + 1) & \text{on} \quad [-1, 0] \times [0, 1], \text{ and} \\ (y + 1) & \text{on} \quad [0, 1] \times [-1, 0] \end{cases}$
3. linear on sub-triangles: $B(x, y) = \max(0, 1 - \max(|x|, |y|, |x + y|))$

[4] In the remainder of the paper, we will use indices $i$ for pixels and $j$ for spline control vertices.

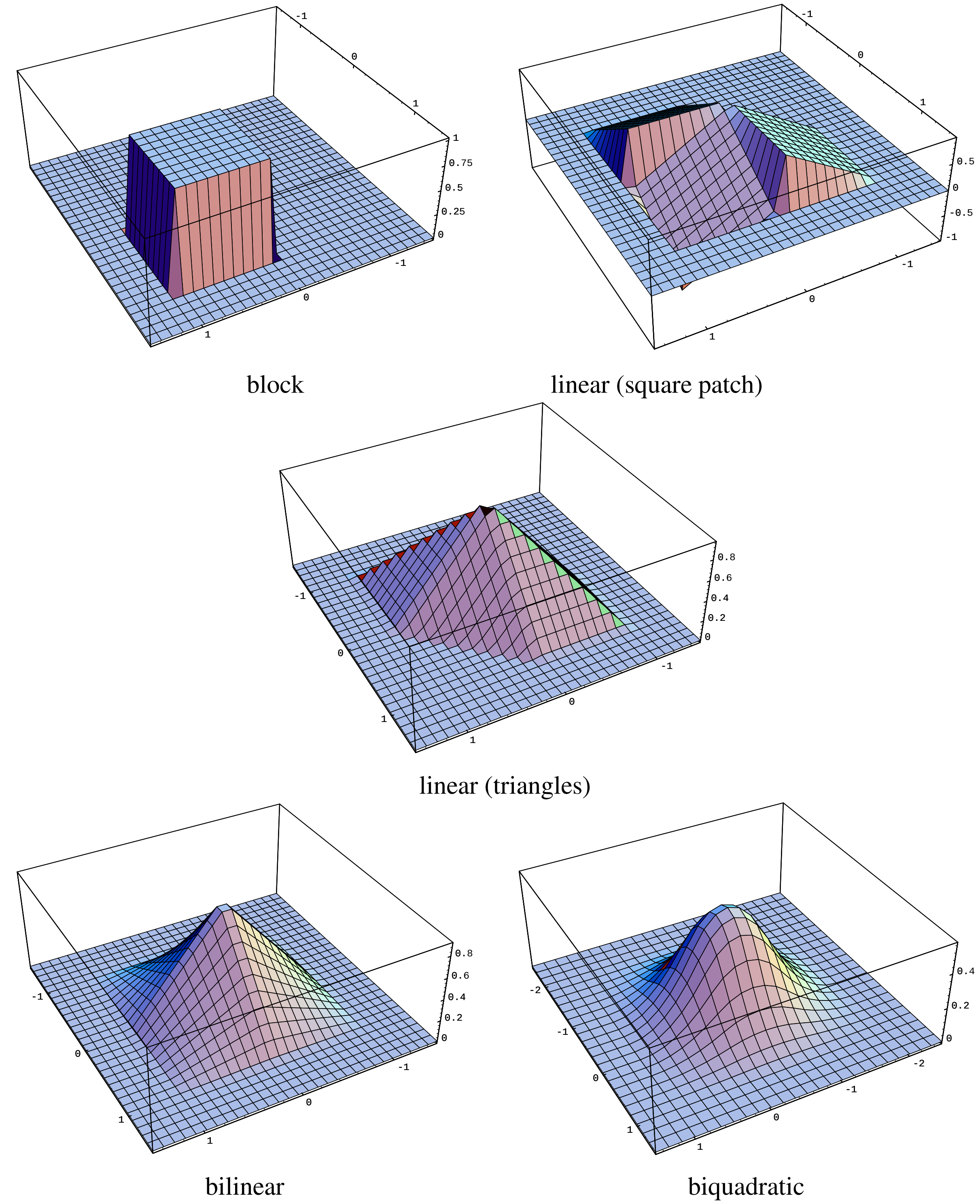



Figure 2: Spline basis functions

4. bilinear: $B(x, y) = (1 - |x|)(1 - |y|)$ on $[-1, 1]^2$

5. biquadratic: $B(x, y) = B_2(x)B_2(y)$ on $[-1, 2]^2$, where $B_2(x)$ is the quadratic B-spline

Figure 2 shows 3D graphs of the basis functions for the five splines. We also impose the condition that the spline control grid is a regular subsampling of the pixel grid, $\hat{x}_j = mx_i$, $\hat{y}_j = my_i$, so that each set of $m \times m$ pixels corresponds to a single spline patch. This means that the set of $w_{ij}$ weights need only be stored for a single patch.

How do spline representations compare to local correlation windows and to regularization? This question has previously been studied in the context of active deformable contours (*snakes*). The original work on snakes was based on a regularization framework [Kass *et al.*, 1988], giving the snake the ability to model arbitrarily detailed bends or discontinuities where warranted by the data. More recent versions of snakes often employ the *B-snake* [Menet *et al.*, 1990; Blake *et al.*, 1993] which has fewer control vertices. A spline-based snake has fewer degrees of freedom, and thus may be easier to recover. The smooth interpolation function between vertices plays a similar role to regularization, although the smoothing introduced is not as uniform or stationary.

In our use of splines for modeling displacement fields, we have a similar tradeoff (see Section 9 for more discussion). We often may not need regularization (e.g., in highly textured scenes). Where required, adding a regularization term to the cost function (3) is straightforward, i.e., we can use a first order regularizer [Poggio *et al.*, 1985]

$$E_1(\{\hat{u}_{k,l}, \hat{v}_{k,l}\}) = \sum_{k,l} (\hat{u}_{k,l} - \hat{u}_{k-1,l})^2 + (\hat{u}_{k,l} - \hat{u}_{k,l-1})^2 + \cdots \text{ terms in } \hat{v}_{kl} \tag{5}$$

or a second order regularizer [Terzopoulos, 1986]

$$\begin{aligned} E_2(\{\hat{u}_{k.l}, \hat{v}_{k,l}\}) \quad = \quad & h^{-2} \sum_{k,l} (\hat{u}_{k+1,l} - 2\hat{u}_{k,l} + \hat{u}_{k-1,l})^2 + (\hat{u}_{k,l+1} - 2\hat{u}_{k,l} + \hat{u}_{k,l-1})^2 + \\ & (\hat{u}_{k,l} - \hat{u}_{k,l-1} - \hat{u}_{k-1,l} + \hat{u}_{k-1,l-1})^2 + \cdots \text{ terms in } \hat{v}_{kl} \end{aligned} \tag{6}$$

where $h$ is the patch size, and we index the spline control vertices with 2D indices $(k, l)$.

Spline-based flow descriptors also remove the need for overlapping correlation windows, since each flow estimate $(\hat{u}_j, \hat{v}_j)$ is based on weighted contributions from all of the pixels beneath the support of its basis function (e.g, $(2m) \times (2m)$ pixels for a bilinear basis). As we will show in Section 4, the spline-based flow formulation makes it straightforward to compute the uncertainty (covariance matrix) associated with the complete flow field. It also corresponds naturally to the optimal Bayesian estimator for the flow, where the squared pixel errors correspond to Gaussian noise, while the spline model (and any associated regularizers) form the prior model[5] [Szeliski, 1989].

Before moving on to our different motion models and solution techniques, we should point out that the squared pixel error function (3) can be generalized to account for photometric variation (global brightness and contrast changes). Following [Lucas, 1984; Gennert, 1988], we can write

$$E'(\{u_i, v_i\}) = \sum_i [I_1(x_i + u_i, y_i + v_i) - cI_0(x_i, y_i) + b]^2, \tag{7}$$

where $b$ and $c$ are the (per-frame) brightness and contrast correction terms. Both of these parameters can be estimated concurrently with the flow field at little additional cost. Their inclusion is most useful in situations where the photometry can change between successive views (e.g., when the images are not acquired concurrently). We should also mention that the matching need not occur directly on the raw intensity images. Both linear (e.g., low-pass or band-pass filtering [Burt and Adelson, 1983]) and non-linear pre-processing can be performed.

# 4 Local (general) flow estimation

To recover the local spline-based flow parameters, we need to minimize the cost function (3) with respect to the $\{\hat{u}_j, \hat{v}_j\}$. We do this using a variant of the Levenberg-Marquardt iterative non-linear minimization technique [Press *et al.*, 1992]. First, we compute the gradient of $E$ in (3) with respect to each of the parameters $\hat{u}_j$ and $\hat{v}_j$,

$$\begin{aligned} g_j^u &\equiv \frac{\partial E}{\partial \hat{u}_j} = 2\sum_i e_i G_i^x w_{ij} \\ g_j^v &\equiv \frac{\partial E}{\partial \hat{v}_j} = 2\sum_i e_i G_i^y w_{ij}, \end{aligned} \tag{8}$$

where

$$e_i = I_1(x_i + u_i, y_i + v_i) - I_0(x_i, y_i) \tag{9}$$

is the intensity error at pixel $i$,

$$(G_i^x, G_i^y) = \nabla I_1(x_i + u_i, y_i + v_i) \tag{10}$$

[5]Correlation-based techniques with overlapping windows do not have a similar direct connection to Bayesian techniques.

is the intensity gradient of $I_1$ at the displaced position for pixel $i$, and the $w_{ij}$ are the sampled values of the spline basis function (4). Algorithmically, we compute the above gradients by first forming the displacement vector for each pixel $(u_i, v_i)$ using (4), then computing the resampled intensity and gradient values of $I_1$ at $(x'_i, y'_i) = (x_i + u_i, y_i + v_i)$, computing $e_i$, and finally incrementing the $g_j^u$ and $g_j^v$ values of all control vertices affecting that pixel.

For the Levenberg-Marquardt algorithm, we also require the approximate Hessian matrix $\mathbf{A}$ where the second-derivative terms are left out.[6] The matrix $\mathbf{A}$ contains entries of the form

$$\begin{aligned} a_{jk}^{uu} &= 2\sum_i \frac{\partial e_i}{\partial \hat{u}_j}\frac{\partial e_i}{\partial \hat{u}_k} = 2\sum_i w_{ij}w_{ik}(G_i^x)^2 \\ a_{jk}^{uv} = a_{jk}^{vu} &= 2\sum_i \frac{\partial e_i}{\partial \hat{u}_j}\frac{\partial e_i}{\partial \hat{v}_k} = 2\sum_i w_{ij}w_{ik}G_i^x G_i^y \\ a_{jk}^{vv} &= 2\sum_i \frac{\partial e_i}{\partial \hat{v}_j}\frac{\partial e_i}{\partial \hat{v}_k} = 2\sum_i w_{ij}w_{ik}(G_i^y)^2. \end{aligned} \tag{11}$$

The entries of $\mathbf{A}$ can be computed at the same time as the energy gradients.

What is the structure of the approximate Hessian matrix? The $2 \times 2$ sub-matrix $\mathbf{A}_{jj}$ corresponding to the terms $a_{jj}^{uu}$, $a_{jj}^{uv}$, and $a_{jj}^{vv}$ encodes the local shape of the sum-of-squared difference correlation surface [Lucas, 1984; Anandan, 1989]. This matrix is often used to compute an updated flow vector by setting

$$[\Delta\hat{u}_j \;\; \Delta\hat{v}_j]^T = -\mathbf{A}_{jj}^{-1}[g_j^u \;\; g_j^v]^T \tag{12}$$

[Lucas, 1984; Anandan, 1989; Bergen *et al.*, 1992]. The overall $\mathbf{A}$ matrix is a sparse multi-banded block-diagonal matrix, i.e., sub-blocks $\mathbf{A}_{jk}$ will be non-zero only if vertices $j$ and $k$ both influence some common patch of pixels.

The Levenberg-Marquardt algorithm proceeds by computing an increment $\Delta\mathbf{u}$ to the current displacement estimate $\mathbf{u}$ which satisfies

$$(\mathbf{A} + \lambda\mathrm{diag}(\mathbf{A}))\Delta\mathbf{u} = -\mathbf{g}, \tag{13}$$

where $\mathbf{u}$ is the vector of concatenated displacement estimates $\{\hat{u}_j, \hat{v}_j\}$, $\mathbf{g}$ is the vector of concatenated energy gradients $\{g_j^u, g_j^v\}$, and $\lambda$ is a stabilization factor which varies over time [Press *et al.*, 1992]. For systems with small numbers of parameters, e.g., if only a single spline patch is

[6] As mentioned in [Press *et al.*, 1992], inclusion of these terms can be destabilizing if the model fits badly or is contaminated by outlier points.

being used (Section 5), this system of equations can be solved at reasonable computational cost. However, for general flow computation, there may be thousands of spline control variables (e.g., for a 640 $\times$ 480 image with $m = 8$, we have $81 \times 61 \times 2 \approx 10^4$ parameters). In this case, iterative sparse matrix techniques have to be used to solve the above system of equations.[7]

In our current implementation, we use preconditioned gradient descent to update our flow estimates

$$\Delta\mathbf{u} = -\alpha\,\mathbf{B}^{-1}\mathbf{g} = -\alpha\hat{\mathbf{g}} \tag{14}$$

where $\mathbf{B} = \hat{\mathbf{A}} + \lambda\mathbf{I}$, and $\hat{\mathbf{A}} = \text{block\_diag}(\mathbf{A})$ is the set of $2 \times 2$ block diagonal matrices used in (12).[8] In this simplest version, the update rule is very close to that used by [Lucas, 1984] and others, with the following differences:

1. the equations for computing the $\mathbf{g}$ and $\mathbf{A}$ are different (based on spline interpolation)
2. an additional diagonal term $\lambda$ is added for stability[9]
3. there is a step size $\alpha$.

The step size $\alpha$ is necessary because we are ignoring the off-block-diagonal terms in $\mathbf{A}$, which can be quite significant. An optimal value for $\alpha$ can be computed at each iteration by minimizing

$$\Delta E(\alpha\mathbf{d}) \approx \alpha^2 \mathbf{d}^T\mathbf{A}\mathbf{d} - 2\alpha\mathbf{d}^T\mathbf{g},$$

i.e., by setting $\alpha = (\mathbf{d}\cdot\mathbf{g})/(\mathbf{d}^T\mathbf{A}\mathbf{d})$. The denominator can be computed without explicitly computing $\mathbf{A}$ by noting that

$$\mathbf{d}^T\mathbf{A}\mathbf{d} = \sum_i (G_i^x \delta u_i + G_i^y \delta v_i)^2 \;\text{ where }\; \delta u_i = \sum_j w_{ij}\delta\hat{u}_j, \;\; \delta v_i = \sum_j w_{ij}\delta\hat{v}_j,$$

and the $(\delta\hat{u}_j, \delta\hat{v}_j)$ are the components of $\mathbf{d}$.

To handle larger displacements, we run our algorithm in a coarse-to-fine (hierarchical) fashion. A Gaussian image pyramid is first computed using an iterated 3-pt filter [Burt and Adelson,

[7] Excessive *fill in* prevents the application of direct sparse matrix solvers [Terzopoulos, 1986; Szeliski, 1989].

[8] The vector $\hat{\mathbf{g}} = \mathbf{B}^{-1}\mathbf{g}$ is called the *preconditioned residual vector*. For preconditioned conjugate gradient descent, the *direction vector* $\mathbf{d}$ is set to $\hat{\mathbf{g}}$.

[9] A Bayesian justification can be found in [Simoncelli *et al.*, 1991], and additional possible local weightings in [Lucas, 1984, p. 20].

1983]. We then run the algorithm on one of the smaller pyramid levels, and use the resulting flow estimates to initialize the next finer level (using bilinear interpolation and doubling the displacement magnitudes). Figure 3 shows a block diagram of the processing stages involved in our spline-based image registration algorithm.

Figure 4 shows an example of the flow estimates produced by our technique. The input image is $256 \times 240$ pixels, and the flow is displayed on a $30 \times 28$ grid. We used $16 \times 16$ pixel spline patches, and 3 levels in the pyramid, with 9 iterations at each level. The flow estimates are very good in the textured areas corresponding to the Rubik cube, the stationary boxes, and the turntable edges. Flow vectors in the uniform intensity areas (e.g., table and turntable tops) are fairly arbitrary. This example uses no regularization beyond that imposed by the spline patches, nor does it threshold flow vectors according to certainty. For a more detailed analysis, see Section 8.

# 5 Global (planar) flow estimation

In many applications, e.g., in the registration of pieces of a flat scene, when the distance between the camera and the scene is large [Bergen *et al.*, 1992], or when performing a coarse registration of slices in a volumetric data set [Carlbom *et al.*, 1991], a single global description of the motion model may suffice. A simple example of such a global motion is an affine flow [Koenderink and van Doorn, 1991; Rehg and Witkin, 1991; Bergen *et al.*, 1992]

$$\begin{aligned} u(x,y) &= (m_0 x + m_1 y + m_2) - x \\ v(x,y) &= (m_3 x + m_4 y + m_5) - y. \end{aligned} \tag{15}$$

The parameters $\mathbf{m} = (m_0, \ldots, m_5)^T$ are called the *global motion parameters*. Models with fewer degrees of freedom such as pure translation, translation and rotation, or translation plus rotation and scale (*similarity transform*) are also possible, but they will not be studied in this paper.

To compute the global motion estimate, we take a two step approach. First, we define the spline control vertices $\hat{\mathbf{u}}_j = (\hat{u}_j, \hat{v}_j)^T$ in terms of the global motion parameters

$$\hat{\mathbf{u}}_j = \begin{bmatrix} \hat{x}_j & \hat{y}_j & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \hat{x}_j & \hat{y}_j & 1 \end{bmatrix} \mathbf{m} - \begin{bmatrix} \hat{x}_j \\ \hat{y}_j \end{bmatrix} \equiv \mathbf{T}_j \mathbf{m} - \hat{\mathbf{x}}_j. \tag{16}$$

Second, we define the flow at each pixel using our usual spline interpolation. Note that for affine

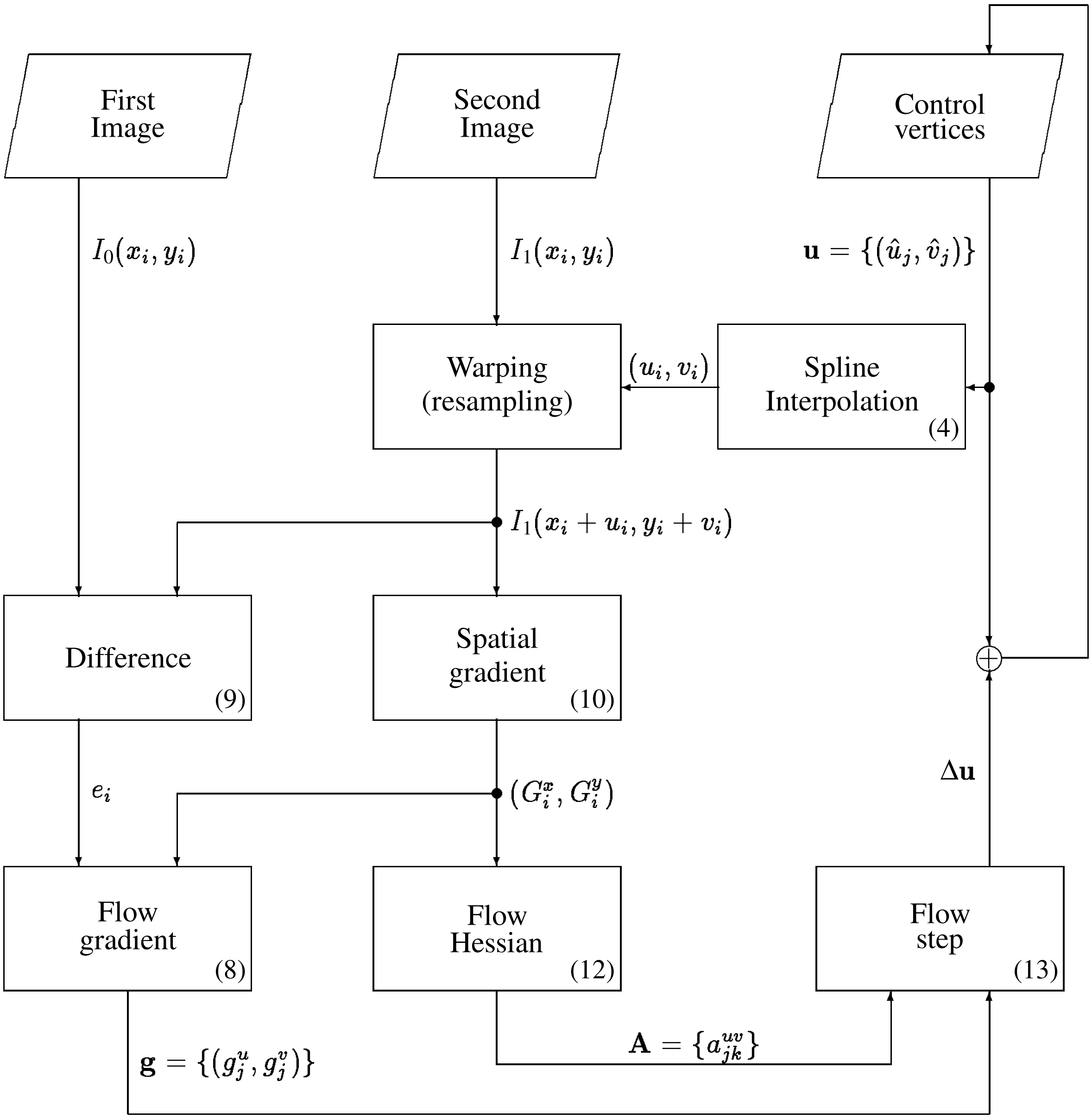
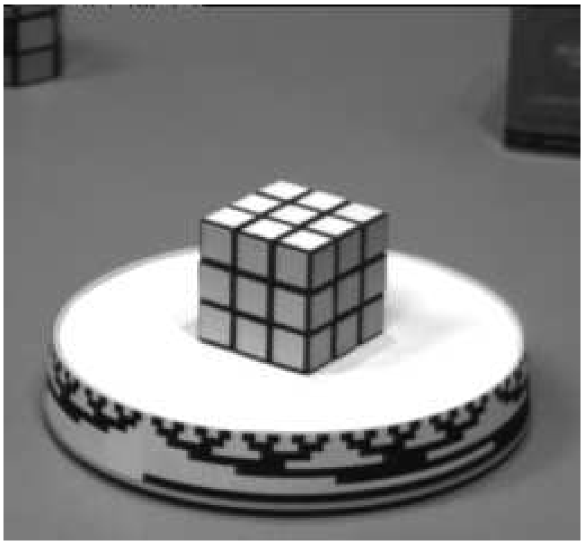



Figure 3: Block diagram of spline-based image registration
The numbers in the lower right corner of each processing box refer to the associated equation numbers in the paper.



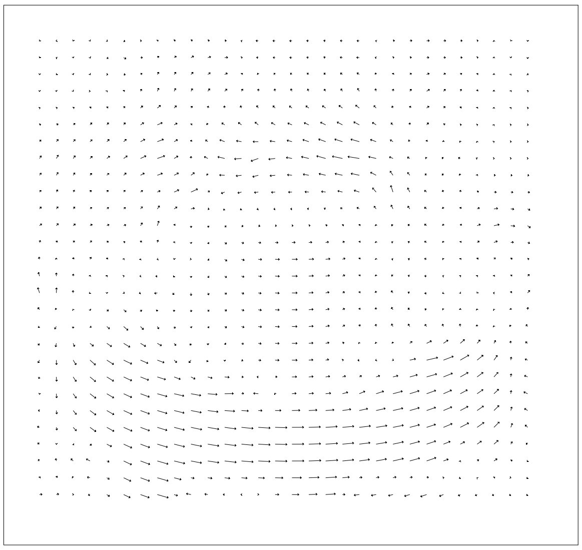


Figure 4: Example of general flow computation

(or simpler) flow, this gives the correct flow at each pixel if linear or bilinear interpolants are used.[10] For affine (or simpler) flow, it is therefore possible to use only a single spline patch.[11]

Why use this two-step procedure then? First, this approach will work better when we generalize our motion model to 2D projective transformations (see below). Second, there are computational savings in only storing the $w_{ij}$ for smaller patches. Lastly, we can obtain a better estimate of the Hessian at $\mathbf{m}$ at lower computational cost, as we discuss below.

To apply Levenberg-Marquardt as before, we need to compute both the gradient of the cost function with respect to $\mathbf{m}$ and the Hessian. Computing the gradient is straightforward

$$\mathbf{g}_{\mathrm{m}} \equiv \frac{\partial E}{\partial \mathbf{m}} = \sum_j \frac{\partial \hat{\mathbf{u}}_j}{\partial \mathbf{m}} \frac{\partial E}{\partial \hat{\mathbf{u}}_j} = \sum_j \mathbf{T}_j^T \mathbf{g}_j \tag{17}$$

where $\mathbf{g}_j = (g_j^u, g_j^v)^T$. The Hessian matrix can be computed in a similar fashion

$$\mathbf{A}_{\mathrm{m}} = \frac{\partial^2 E}{\partial \mathbf{m}^T \partial \mathbf{m}} \approx \sum_{jk} \mathbf{T}_j^T \mathbf{A}_{jk} \mathbf{T}_k, \tag{18}$$

where the $\mathbf{A}_{jk}$ are the $2 \times 2$ submatrices of $\mathbf{A}$.

[10] The definitions of $\hat{u}_j$ and $\hat{v}_j$ would have to be adjusted if biquadratic splines are being used.

[11] Multiple affine patches can also be used to track features [Rehg and Witkin, 1991]. Our general flow estimator (Section 4) performs a similar function, except in parallel and with overlapping windows.

We can approximate the Hessian matrix even further by neglecting the off-diagonal $\mathbf{A}_{jk}$ matrices. This is equivalent to modeling the flow estimate at each control vertex as being independent of other vertex estimates. When the spline patches are sufficiently large and contain sufficient texture, this turns out to be a reasonable approximation.

To compute the optimal step size $\alpha$, we let

$$\mathbf{d} = \mathbf{T}\mathbf{d}_{\mathrm{m}} \tag{19}$$

where $\mathbf{T}$ is the concatenation of all the $\mathbf{T}_j$ matrices, and then set $\alpha = (\mathbf{d} \cdot \mathbf{g})/(\mathbf{d}^T \mathbf{A}\mathbf{d})$ as before. Figure 5 shows a block diagram of the processing stages involved in the global motion estimation algorithms presented in this and the next section.

Examples of our global affine flow estimator applied to two different motion sequences can be seen in Figures 6 and 7. These two sequences were generated synthetically from a real base image, with Figure 6 being pure translational motion, and Figure 7 being divergent motion [Barron *et al.*, 1994]. As expected, the motion is recovered extremely well in this case (see Section 8 for quantitative results).

A more interesting case, in general, is that of a planar surface in motion viewed through a pinhole camera. This motion can be described as a 2D projective transformation of the plane

$$\begin{aligned} u(x,y) &= \frac{m_0 x + m_1 y + m_2}{m_6 x + m_7 y + \ 1} - x \\ v(x,y) &= \frac{m_3 x + m_4 y + m_5}{m_6 x + m_7 y + \ 1} - y. \end{aligned} \tag{20}$$

Our projective formulation[12] requires 8 parameters per frame, which is the same number as the quadratic flow field used in [Bergen *et al.*, 1992]. However, our formulation allows for arbitrarily large displacements, whereas [Bergen *et al.*, 1992] is based on instantaneous (infinitesimal) motion. Our formulation also does not require the camera to be calibrated, and allows the internal camera parameters (e.g., zoom) to vary over time. The price we pay is that the motion field is no longer a linear function of the global motion parameters.

To compute the gradient and the Hessian, we proceed as before. We use the equations

$$\hat{u}_j = \frac{m_0 \hat{x}_j + m_1 \hat{y}_j + m_2}{m_6 \hat{x}_j + m_7 \hat{y}_j + \ 1} - \hat{x}_j$$

[12] In its full generality, we should have an $m_8$ instead of a 1 in the denominator. The situation $m_8 = 0$ occurs only for 90° camera rotations.

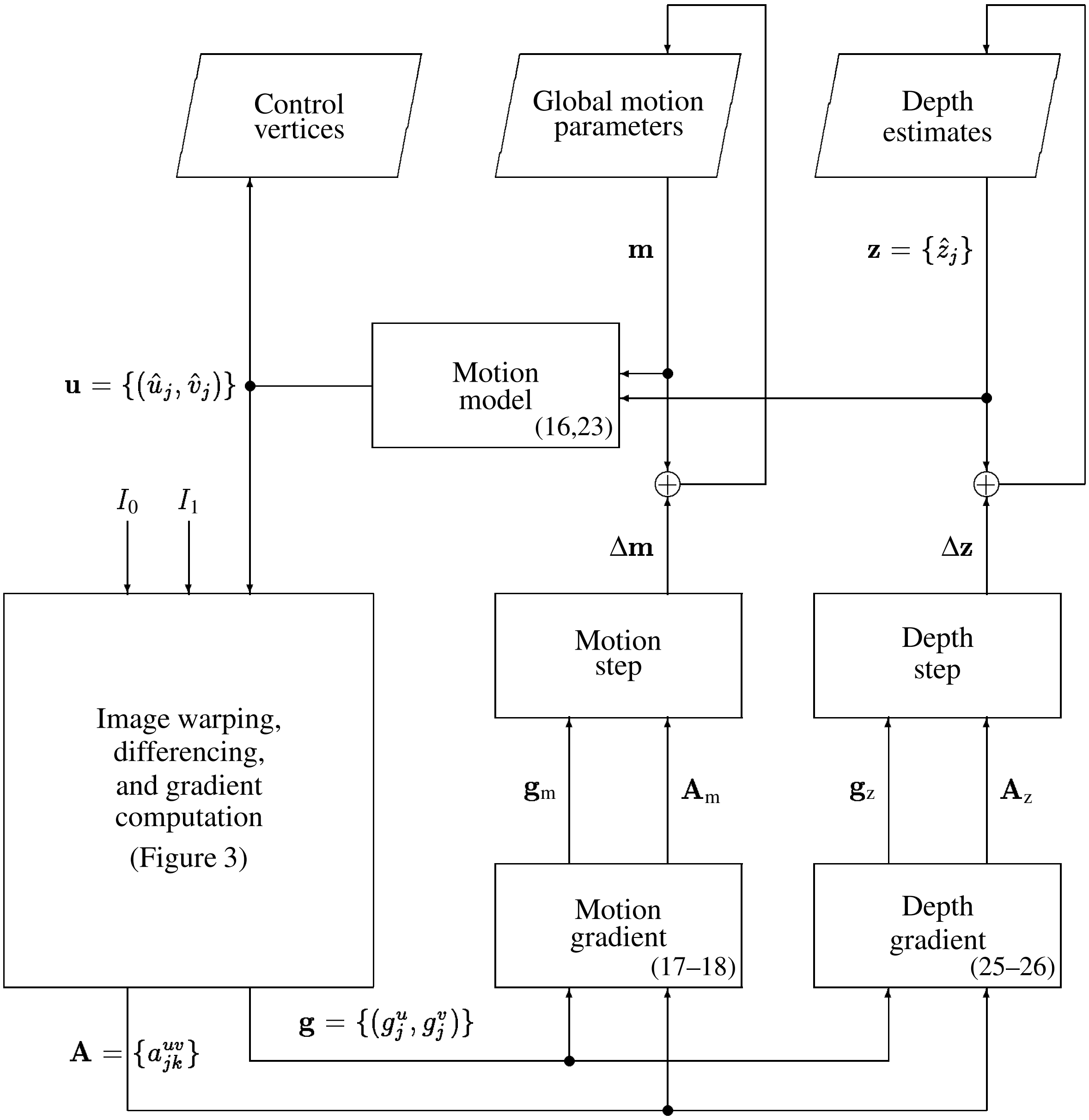



Figure 5: Block diagram of global motion estimation

The numbers in the lower right corner of each processing box refer to the associated equation numbers in the paper.

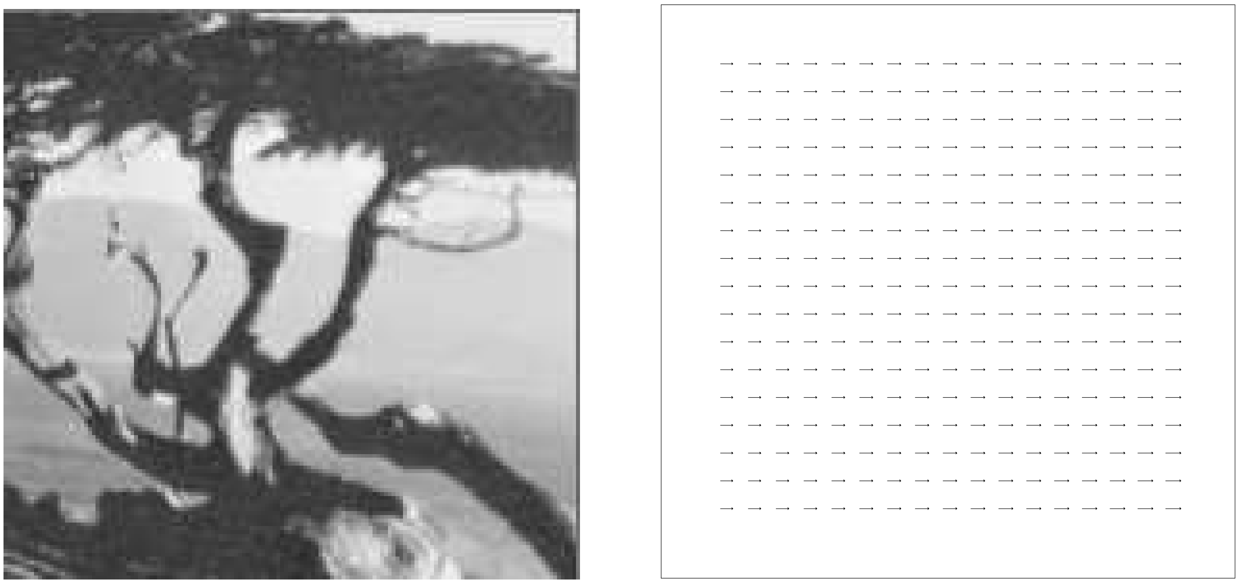


Figure 6: Example of affine flow computation: translation

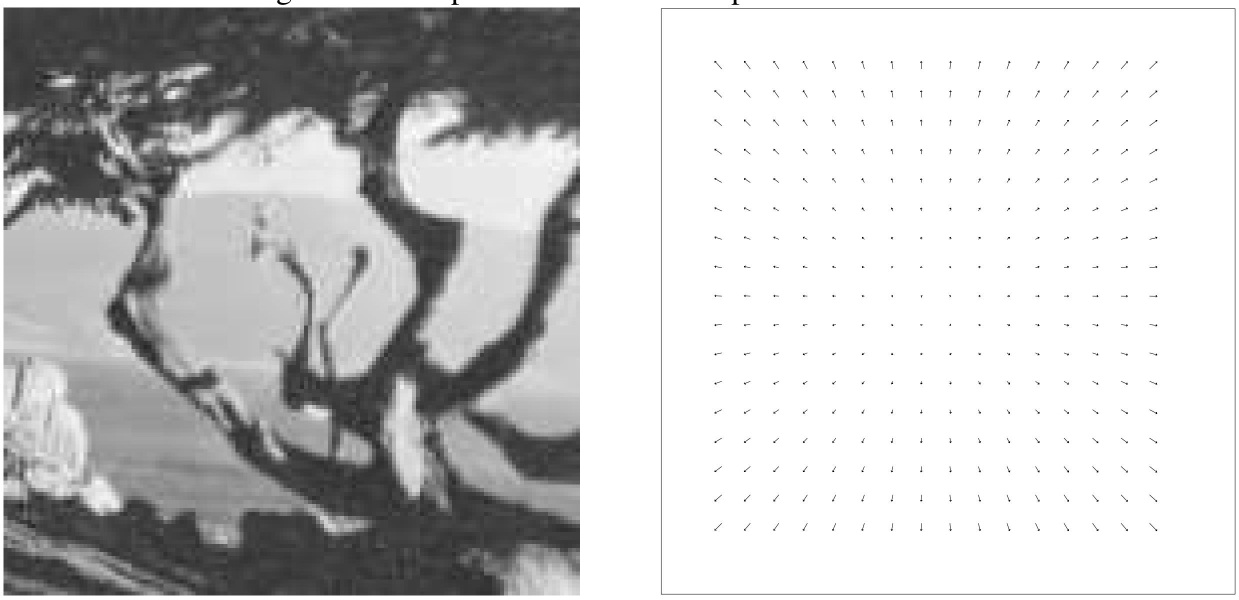


Figure 7: Example of affine flow computation: divergence (zoom)

$$\hat{v}_j = \frac{m_3\hat{x}_j + m_4\hat{y}_j + m_5}{m_6\hat{x}_j + m_7\hat{y}_j + 1} - \hat{y}_j \tag{21}$$

to compute the spline control vertices, and use the B-spline interpolants to compute the flow at each pixel. This flow is not exactly equivalent to the true projective flow defined in (20), since the latter involves a division at each pixel. However, the error will be small if the patches are small and/or the perspective distortion $(m_6, m_7)$ is small.

To compute the gradient, we note that

$$\delta\hat{\mathbf{u}}_j = \frac{1}{D_j}\begin{bmatrix} \hat{x}_j & \hat{y}_j & 1 & 0 & 0 & 0 & -\hat{x}_j N_j^u/D_j & -\hat{y}_j N_j^u/D_j \\ 0 & 0 & 0 & \hat{x}_j & \hat{y}_j & 1 & -\hat{x}_j N_j^v/D_j & -\hat{y}_j N_j^v/D_j \end{bmatrix} \delta\mathbf{m} \equiv \mathbf{T}_j \delta\mathbf{m} \tag{22}$$

where $N_j^u = m_0\hat{x}_j + m_1\hat{y}_j + m_2$ and $N_j^v = m_3\hat{x}_j + m_4\hat{y}_j + m_5$ are the current numerators of (21), $D_j = m_6\hat{x}_j + m_7\hat{y}_j + 1$ is the current denominator, and $\mathbf{m} = (m_0, \ldots, m_7)^T$. With this modification to the affine case, we can proceed as before, applying (17)–(19) to compute the global gradient, Hessian, and the stepsize. Thus, we see that even though the problem is no longer linear, the modifications involve simply using an extra division per spline control vertex. Figure 8 shows an image which has been perspectively distorted and the accompanying recovered flow field. The motion is that of a plane rotating around its $y$ axis and moving slightly forward, viewed with a wide angle lens.

# 6 Mixed global and local (rigid) flow estimation

A special case of optic flow computation which occurs often in practice is that of *rigid motion*, i.e., when the camera moves through a static scene, or a single object moves rigidly in front of a camera. Commonly used techniques (*direct methods*) based on estimating the instantaneous camera egomotion $(\mathbf{R}(\omega), \mathbf{t})$ and a camera-centered depth $Z(x, y)$ are given in [Horn and Weldon Jr., 1988; Hanna, 1991; Bergen *et al.*, 1992]. This has the disadvantage of only being valid for small motions, of requiring a calibrated camera, and of sensitivity problems with the depth estimates.[13]

Our approach is based on a *projective* formulation of structure from motion [Hartley *et al.*, 1992; Faugeras, 1992; Mohr *et al.*, 1993; Szeliski and Kang, 1994]

$$u(x, y) = \frac{m_0 x + m_1 y + m_8\, z(x, y) + m_2}{m_6 x + m_7 y + m_{10} z(x, y) + 1} - x$$

[13][Bergen *et al.*, 1992] mitigate the $Z$ sensitivity problem by estimating $1/Z(x, y)$ instead.

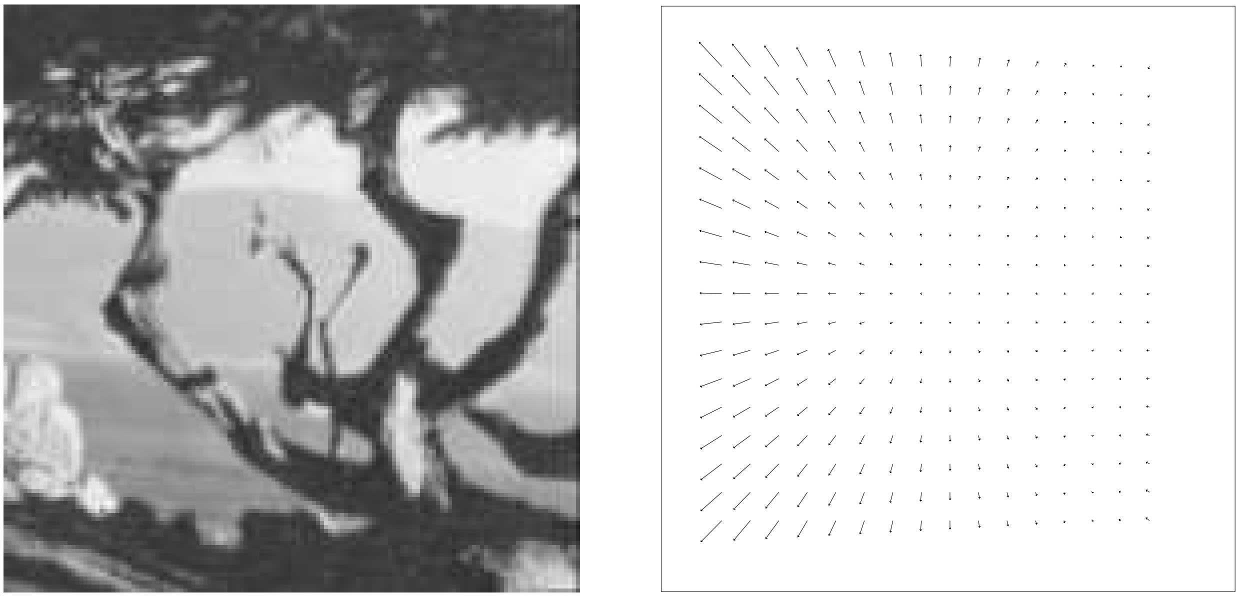


Figure 8: Example of 2D projective motion estimation

$$v(x,y) = \frac{m_3 x + m_4 y + m_9\, z(x,y) + m_5}{m_6 x + m_7 y + m_{10} z(x,y) + 1} - y. \quad (23)$$

This formulation is valid for any pinhole camera model, even with time varying internal camera parameters. The local shape estimates $z(x,y)$ are *projective depth* estimates, i.e., the $(x,y,z,1)$ coordinates are related to the true Euclidean coordinates $(X,Y,Z,1)$ through some 3-D projective transformation (*collineation*) which can, given enough views, be recovered from the projective motion estimates [Szeliski, 1994b].[14]

Compared to the usual rigid motion formulation, we have to estimate more global parameters (11 instead of 6) for the global motion, so we might be concerned with an increased uncertainty in these parameters. However, we do not require our camera to be calibrated or to have fixed internal parameters. We can also deal with arbitrarily large displacements and non-smooth motion. Furthermore, situations in which either the global motion or local shape estimates are poorly recovered (e.g., planar scenes, pure rotation) do not cause any problems for our technique.

A special case of the rigid motion problem is when we are given a pair of calibrated images

[14]There is an ambiguity in the shape recovered, i.e., we can add any plane equation to the $z$ coordinates, $z' = ax + by + cz + d$, and still have a valid solution after adjusting the $m_l$. This ambiguity is not a problem for gradient descent techniques.

(*stereo*) or a set of *weakly calibrated* images (only the epipoles in the images are known [Hartley *et al.*, 1992; Faugeras, 1992]). In either case, we can compute the $m_l$, resulting in a simple estimation problem in $z(x, y)$. We can then either proceed to resample (*rectify*) the images so that corresponding pixels lie along scanlines [Hartley and Gupta, 1993], or we can directly work with the original images, proceeding as below but omitting the $\mathbf{m}$ update step.

To compute the global and local flow estimates, we combine several of the approaches developed previously in the paper. First, we compute the 2D flows at the control vertices by evaluating (23) at the vertex locations $\{(\hat{x}_j, \hat{y}_j)\}$. We compute the gradients and Hessian with respect to the global motion parameters as before, with

$$\mathbf{T}_j = \frac{1}{D_j}\begin{bmatrix} \hat{x}_j & \hat{y}_j & 1 & 0 & 0 & 0 & -\hat{x}_j N_j^u/D_j & -\hat{y}_j N_j^u/D_j & \hat{z}_j & 0 & -\hat{z}_j N_j^u/D_j \\ 0 & 0 & 0 & \hat{x}_j & \hat{y}_j & 1 & -\hat{x}_j N_j^v/D_j & -\hat{y}_j N_j^v/D_j & 0 & \hat{z}_j & -\hat{z}_j N_j^v/D_j \end{bmatrix}, \tag{24}$$

$N_j^u = m_0\hat{x}_j + m_1\hat{y}_j + m_8\hat{z}_j + m_2$, $N_j^v = m_3\hat{x}_j + m_4\hat{y}_j + m_9\hat{z}_j + m_5$, and $D_j = m_6\hat{x}_j + m_7\hat{y}_j + m_{10}\hat{z}_j + 1$. The derivatives with respect to the depth estimates $\hat{z}_j$ are

$$g_j^z \equiv \frac{\partial E}{\partial \hat{z}_j} = \frac{\partial E}{\partial \hat{u}_j}\frac{\partial \hat{u}_j}{\partial \hat{z}_j} + \frac{\partial E}{\partial \hat{v}_j}\frac{\partial \hat{u}_j}{\partial \hat{z}_j} = g_j^u \frac{m_8 - m_{10}N_j^u/D_j}{D_j} + g_j^v \frac{m_9 - m_{10}N_j^v/D_j}{D_j} \tag{25}$$

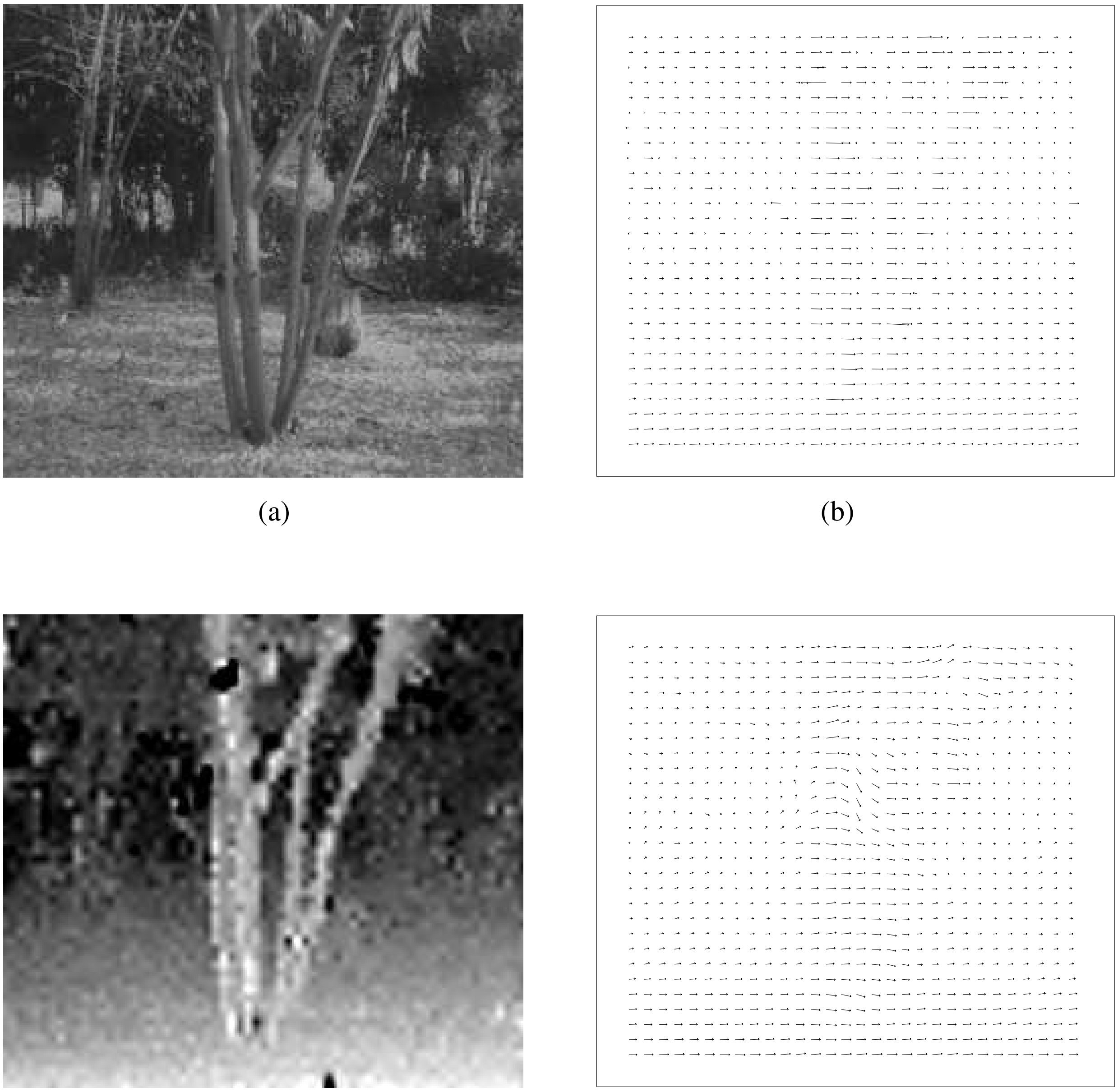
The Hessian matrix $\mathbf{A}_z$ for the $\mathbf{z} = \{\hat{z}_j\}$ parameters has components

$$a_{jk}^z = (\mathbf{p}_j^{uv/z})^T \mathbf{A}_{jk}\mathbf{p}_k^{uv/z} \quad \text{with} \quad \mathbf{p}_j^{uv/z} = (p_j^{u/z}, p_j^{v/z})^T \equiv (\frac{\partial \hat{u}_j}{\partial \hat{z}_j}, \frac{\partial \hat{v}_j}{\partial \hat{z}_j})^T. \tag{26}$$

There are two ways at this point to estimate the $\mathbf{m}$ and $\mathbf{z}$ parameters. We can take simultaneous steps in $\Delta\mathbf{m}$ and $\Delta\mathbf{z}$, or we can alternate steps in $\Delta\mathbf{m}$ and $\Delta\mathbf{z}$. The former approach is the one we adopted in [Szeliski, 1994b], since the full Hessian matrix had already been computed thus enabling a regular Levenberg-Marquardt step, and this proved to have faster convergence. In this work, we adopt the latter approach, since it proved to be simpler to implement. We plan to compare both approaches in future work.

The performance of our rigid motion estimation algorithm on a sample image sequence is shown in Figure 9. As can be seen, the overall direction of motion (the *epipolar geometry*) has been recovered well, and the motion estimates look reasonable.[15] The computed depth map is shown in grayscale. The lower right flow field shows the local (general flow) model applied to the same image pair.

[15] We initialized the $\mathbf{m}$ vector in this example to a horizontal epipolar geometry. If unknown, the epipolar geometry can be recovered from a general 2-D motion field [Faugeras, 1992; Hartley *et al.*, 1992; Hartley and Gupta, 1993; Szeliski, 1994b].



(a) (b)

(c) (d)

Figure 9: Example of 3D projective (rigid) motion estimation
(a) intensity image, (b) constrained rigid flow, (c) recovered depth map, (d) unconstrained local flow (for comparison)

# 7 Multiframe flow estimation

Many current optical flow techniques use more than two images to arrive at local estimates of flow. This is particularly true of spatio-temporal filtering approaches [Adelson and Bergen, 1985; Heeger, 1987; Fleet and Jepson, 1990]. For example, the implementation of [Fleet and Jepson, 1990] described in [Barron *et al.*, 1994] uses 21 images per estimate. Stereo matching techniques have also successfully used multiple images [Bolles *et al.*, 1987; Matthies *et al.*, 1989; Okutomi and Kanade, 1993]. Using large numbers of images not only improves the accuracy of the estimates through noise averaging, but it can also disambiguate between possible matches [Okutomi and Kanade, 1993].

The extension of our local, global, and mixed motion models to multiple frames is straightforward. For local flow, we assume that displacements between successive images and a base image are known scalar multiples of each other,

$$u_t(x,y) = s_t u_1(x,y) \quad \text{and} \quad v_t(x,y) = s_t v_1(x,y), \tag{27}$$

i.e., that we have linear flow (no acceleration).[16] We then minimize the overall cost function

$$E(\{u_i, v_i\}) = \sum_t \sum_i [I_t(x_i + s_t u_i, y_i + s_t v_i) - I_0(x_i, y_i)]^2. \tag{28}$$

This approach is similar to the *sum of sum of squared-distance* (SSSD) algorithm of [Okutomi and Kanade, 1993], except that we represent the motion with a subsampled set of spline coefficients, eliminating the need for overlapping correlation windows.

The modifications to the flow estimation algorithm are minor and obvious. For example, the gradient with respect to the local flow estimate $\hat{u}_j$ in (8) becomes

$$g_j^u = 2 \sum_t s_t \sum_i e_{ti} G_{ti}^x w_{ij}$$

with $e_{ti}$ and $G_{ti}^x$ being the same as $e_i$ and $G_i^x$ with $I_1$ replaced by $I_t$. Given a block of images, estimating two sets of flows, one registered with the first image and another registered with the last, would allow us to do bidirectional prediction for motion-compensated video coding [Le Gall, 1991]. Examples of the improvements in accuracy due to multiframe estimation are given in Section 8.

[16] In the most common case, a uniform temporal sampling ($s_t = t$) is assumed, but this is not strictly necessary [Okutomi and Kanade, 1993].

For global motion estimation, we can either assume that the motion estimates $\mathbf{m}_t$ are related by a known transform (e.g., uniform camera velocity), or we can assume an independent motion estimate for each frame. The latter situation seems more useful, especially in multiframe image mosaicing applications. The motion estimation problem in this case decomposes into a set of independent global motion estimation sub-problems.

The multiframe global/local motion estimation problem is more interesting. Here, we can assume that the global motion parameters for each frame $\mathbf{m}_t$ are independent, but that the local shape parameters $\hat{z}_j$ do not vary over time. This is the situation when we analyze multiple arbitrary views of a rigid 3-D scene, e.g., in the multiframe uncalibrated stereo problem. The modifications to the estimation algorithm are also straightforward. The gradients and Hessian with respect to the global motion parameters $\mathbf{m}_t$ are the same as before, except that the denominator $D_{tj}$ is now different for each frame (since it is a function of $\mathbf{m}_t$).

The derivatives with respect to the depth estimates $\hat{z}_j$ are computed by summing over all frames

$$g_j^z = \sum_t p_{tj}^{u/z} g_{tj}^u + p_{tj}^{v/z} g_{tj}^v \tag{29}$$

where the $p_{tj}^{u/z}$ and $p_{tj}^{v/z}$ (which depend on $\mathbf{m}_t$) and the $g_{tj}^u$ and $g_{tj}^v$ (which depend on $I_t$) are different for each frame. Note that we can no longer get away with a single temporally invariant flow field gradient $(g_j^u, g_j^u)$ (another way to see this is that the epipolar lines in each image can be arbitrary).

# 8 Experimental results

In this section, we demonstrate the performance of our algorithms on the standard motion sequences analyzed in [Barron *et al.*, 1994]. Some of the images in these sequences have already been shown in Figures 4–9. The remaining images are shown in Figures 10–14. We follow the organization of [Barron *et al.*, 1994], presenting quantitative results on synthetically generated sequences first, followed by qualitative results on real motion sequences.

Tables 1–5 give the quantitative results of our algorithms. In these tables, the top two rows are copied from [Barron *et al.*, 1994]. The errors are reported as in [Barron *et al.*, 1994], i.e., by converting flow measurements into unit vector in $\mathcal{R}^3$ and taking the angle between them. The density is the percentage of flow estimates reported to have reliable flow estimates. The computation times on a DEC 3000 Model 400 AXP for these algorithms range from 1 second for the $100 \times 100$

| Technique | Average Error | Standard Deviation | Density |
|---|---|---|---|
| Lucas and Kanade (no thresholding) | 2.47° | 0.16° | 100% |
| Fleet and Jepson ($\tau = 1.25$) | 0.03° | 0.01° | 100% |
| local flow ($n = 2, s = 2, L = 1, b = 0$) | 0.17° | 0.02° | 100% |
| local flow ($n = 3, s = 2, L = 1, b = 0$) | 0.07° | 0.01° | 100% |
| local flow ($n = 5, s = 2, L = 1, b = 0$) | 0.03° | 0.01° | 100% |
| local flow ($n = 7, s = 2, L = 1, b = 0$) | 0.02° | 0.01° | 100% |
| affine flow ($s = 2, L = 1, b = 0$) | 0.13° | 0.01° | 100% |
| affine flow ($s = 4, L = 1, b = 0$) | 0.06° | 0.01° | 100% |

Table 1: Summary of **Sinusoid 1** results

**Sinusoid 1** image (single level, 9 iterations) to 30 seconds for the $300 \times 300$ **Nasa Sequence** (three levels, rigid flow, 9 iterations per level).

From the nine algorithms in [Barron *et al.*, 1994], we have chosen to show the Lucas and Kanade results, since their algorithm most closely matches ours and generally gives good results, and the Fleet and Jepson algorithm since it generally gave the best results. The most salient difference between our (local) algorithm and Lucas and Kanade is that we use a spline representation, which removes the need for overlapping correlation windows, and is therefore much more computationally efficient. The biggest difference with Fleet and Jepson is that they use the whole image sequence (20 frames) whereas we normally use only two (multiframe results are shown in Table 3).

As with many motion estimation algorithms, our algorithms require the selection of some relevant parameters. The most important of these are:

- $n$ [2] the number of frames
- $s$ [1] the step between frames, i.e., 1 = consecutive frames, 2 = every other frame, ...
- $m$ [16] the size of the patch (width and height, $m^2$ pixels per patch)
- $L$ [3] the number of coarse-to-fine levels
- $b$ [3] the amount of initial blurring (# of iterations of a box filter)

Unless mentioned otherwise, we used the default values shown in brackets above for the results in Tables 1–5. Bilinear interpolation was used for the flow fields.

The simplest motions to analyze are two constant-translation sequences, **Sinusoid 1** and **Square**

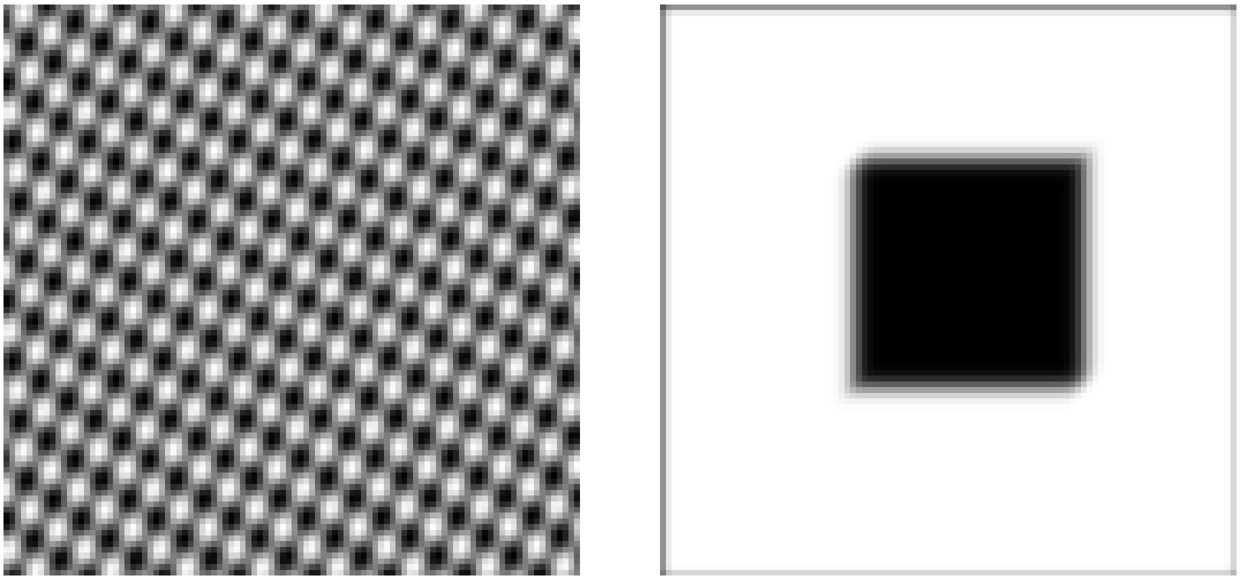


Figure 10: **Sinusoid 1** and **Square 2** sample images

**2** (Figure 10). The translations in these sequences are $(1.585, 0.863)$ and $(1.333, 1.333)$ pixels per frame, respectively. Our local flow estimates for the sinusoid sequence are very good using only two frames (Table 1), and beat all other algorithms when 7 or more frames are used. For this sequence, we use a single level and no blurring and take a $(s = 2)$ frame step for better results. To help overcome local minima for the multiframe ($n > 2$) sequences, we solve a series of easier subproblems [Xu *et al.*, 1987]. We first estimate two-frame motion, then use the resulting estimate to initialize a three-frame estimator, etc. Without this modification, performance on longer (e.g., $n = 8$) sequences would start to degrade because of local minima. The global affine model motion estimator performs well.

For the translating square (Table 2), our results are not as good because of the aperture problem, but with additional regularization, we still outperform all of the nine algorithms studied in [Barron *et al.*, 1994]. To produce the sparse flow estimates (9–23$\%$ density), we set a threshold $T_e$ on the minimum eigenvalues of the local Hessian matrices $\mathbf{A}_{jj}$ interpolated over the whole grid (this selects areas where both components of the motion estimate are well determined). The affine (global) flow for the square sequence works extremely well, outperforming all other techniques by a large margin.

The sequences **Translating Tree** and **Diverging Tree** were generated using a real image (Figure

| Technique | Average Error | Standard Deviation | Density |
|---|---|---|---|
| Lucas and Kanade ($\lambda_2 \geq 5.0$) | 0.14° | 0.10° | 7.9% |
| Fleet and Jepson ($\tau = 2.5$) | 0.18° | 0.13° | 12.6% |
| local flow ($T_e = 10^4$) | 2.98° | 1.16° | 9.1% |
| local flow ($s = 2, T_e = 10^4$) | 1.78° | 1.07° | 10.1% |
| local flow ($s = 2, \lambda_1 = 10^3, T_e = 10^4$) | 0.47° | 0.27° | 23.8% |
| local flow ($s = 2, \lambda_1 = 10^4$) | 0.13° | 0.10° | 100% |
| affine flow | 0.03° | 0.02° | 100% |

Table 2: Summary of **Square 2** results

| Technique | Average Error | Standard Deviation | Density |
|---|---|---|---|
| Lucas and Kanade ($\lambda_2 \geq 5.0$) | 0.56° | 0.58° | 13.1% |
| Fleet and Jepson ($\tau = 1.25$) | 0.23° | 0.19° | 49.7% |
| local flow ($n = 2$) | 0.35° | 0.34° | 100% |
| local flow ($n = 3$) | 0.30° | 0.30° | 100% |
| local flow ($n = 5$) | 0.24° | 0.15° | 100% |
| local flow ($n = 8$) | 0.19° | 0.10° | 100% |
| affine flow | 0.17° | 0.12° | 100% |

Table 3: Summary of **Translating Tree** results

| Technique | Average Error | Standard Deviation | Density |
|---|---|---|---|
| Lucas and Kanade ($\lambda_2 \geq 5.0$) | 1.65° | 1.48° | 24.3% |
| Fleet and Jepson ($\tau = 1.25$) | 0.80° | 0.73° | 46.5% |
| local flow ($s = 4$, $L = 1$) | 0.98° | 0.74° | 100% |
| local flow ($s = 4$, $L = 1$, $\lambda_1 = 10^3$) | 0.78° | 0.47° | 100% |
| affine flow | 2.51° | 0.77° | 100% |

Table 4: Summary of **Diverging Tree** results

| Technique | Average Error | Standard Deviation | Density |
|---|---|---|---|
| Lucas and Kanade ($\lambda_2 \geq 5.0$) | 3.22° | 8.92° | 8.7% |
| Fleet and Jepson ($\tau = 1.25$) | 5.28° | 14.34° | 30.6% |
| local flow ($s = 2$, $T_e = 3000$) | 2.19° | 5.86° | 23.1% |
| local flow ($s = 2$, $T_e = 2000$) | 3.06° | 7.54° | 39.6% |
| local flow, cropped ($s = 2$) | 2.45° | 3.05° | 100% |
| rigid flow, cropped ($s = 2$) | 3.77° | 3.32° | 100% |

Table 5: Summary of **Yosemite** results

7) and synthetic (global) motion. Our results on the translating motion sequence (Table 3) are as good as any other technique for the local algorithm (note the difference in density between our results and the previous ones), and outperform all techniques for the affine motion model, even though we are just using two frames from the sequence. The results on the diverging tree sequence are good for the local flow, but not as good for the affine flow. These results are comparable or better than the other techniques in [Barron *et al.*, 1994] which produce 100% density.

The final motion sequence for which quantitative results are available is **Yosemite** (Figure 11 and Table 5). The images in this sequence were generated by Lynn Quam using his texture mapping algorithm applied to an aerial photograph registered with a digital terrain model. There is significant occlusion and temporal aliasing, and the fractal clouds move independently from the terrain. Our results on this more realistic sequence are better than any of the techniques in [Barron *et al.*, 1994], even though we again only use two images. As expected, the quality of the results depends on the threshold $T_e$ used to produce sparse flow estimates, i.e., there is a tradeoff between

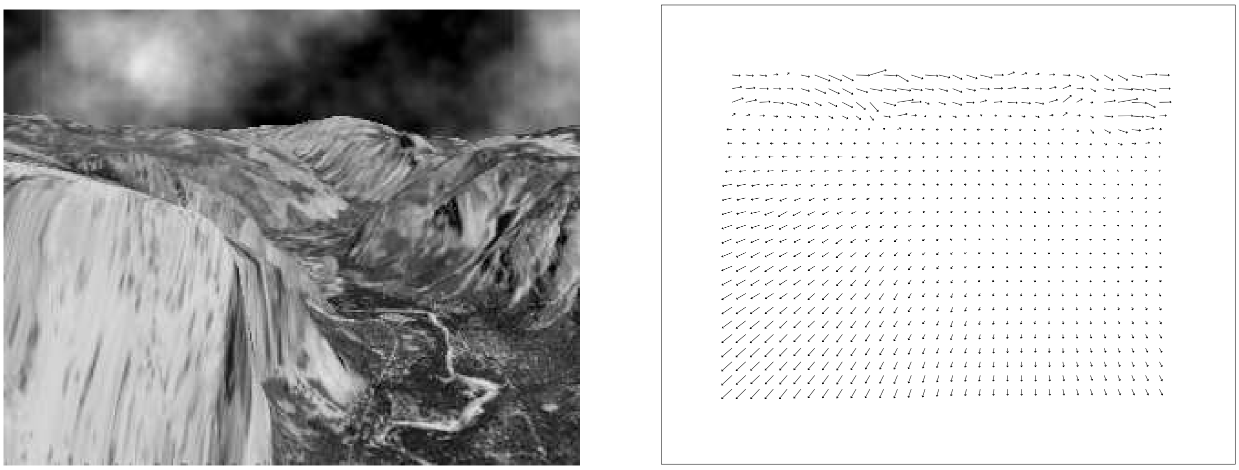


Figure 11: **Yosemite** sample image and flow (unthresholded)

the density of the estimates and their quality. We also ran our algorithm on just the lower 176 (out of 252) rows of the images sequence. The dense (unthresholded) estimates are comparable to the thresholded full-frame estimates. Unfortunately, the results using the rigid motion model were slightly worse.

To conclude our experimental section, we show results on some real motion sequences for which no ground truth data is available. The **SRI Trees** results have already been presented in Figure 9 for both rigid and local (general) flow. Figure 12 shows the **NASA Sequence** in which the camera moves forward in a rigid scene (there is significant aliasing). The motion estimates look quite reasonable, as does the associated depth map (not shown).[17] Figure 13 shows the sparse flow computed for the **Rubik Cube** sequence (the dense flows were shown in Figure 4). The areas with texture and/or corners produce the most reliable flow estimates. Finally, the results on the **Hamburg Taxi** are shown in Figure 14, where the independent motion of the three moving cars can be clearly distinguished. Overall, these results are comparable or better than those shown in [Barron *et al.*, 1994].

Much work remains to be done in the experimental evaluation of our algorithms. In addition to systematically studying the effects of the parameters $n$, $s$, $m$, $L$, and $b$ (introduced previously), we plan to study the effects of different spline interpolation functions, the effects of different preconditioners, and the usefulness of using conjugate gradient descent.

[17] For this sequence and for the Yosemite sequence, we initialized the $\mathbf{m}$ vector to a forward looming motion.

/s.23.pp > /dev/null] --- rigid flow [compute_flow -d64 -n9 -l3 -p -s8 -w3 -y -e8 -Dtmp2.dump -M8 -L0,10,0 -Z1.0 nasa/s.19.pp nasa/s.23.pp >
Image size 300 x 300 Subsampled by 8 scaled by 3.000



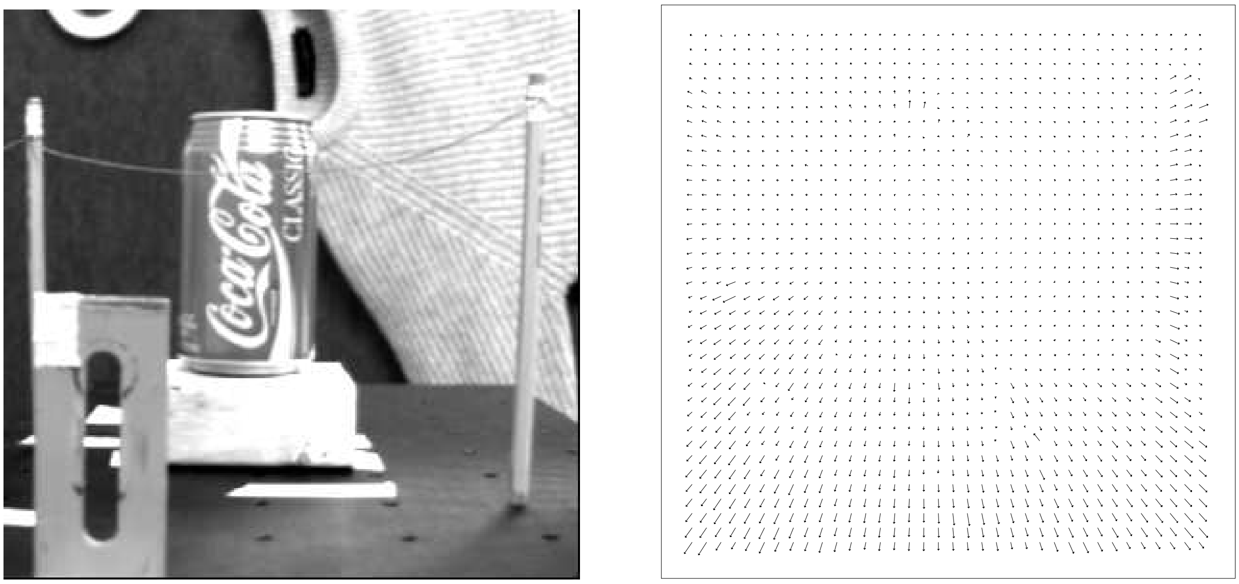



Figure 12: **NASA Sequence**, rigid flow

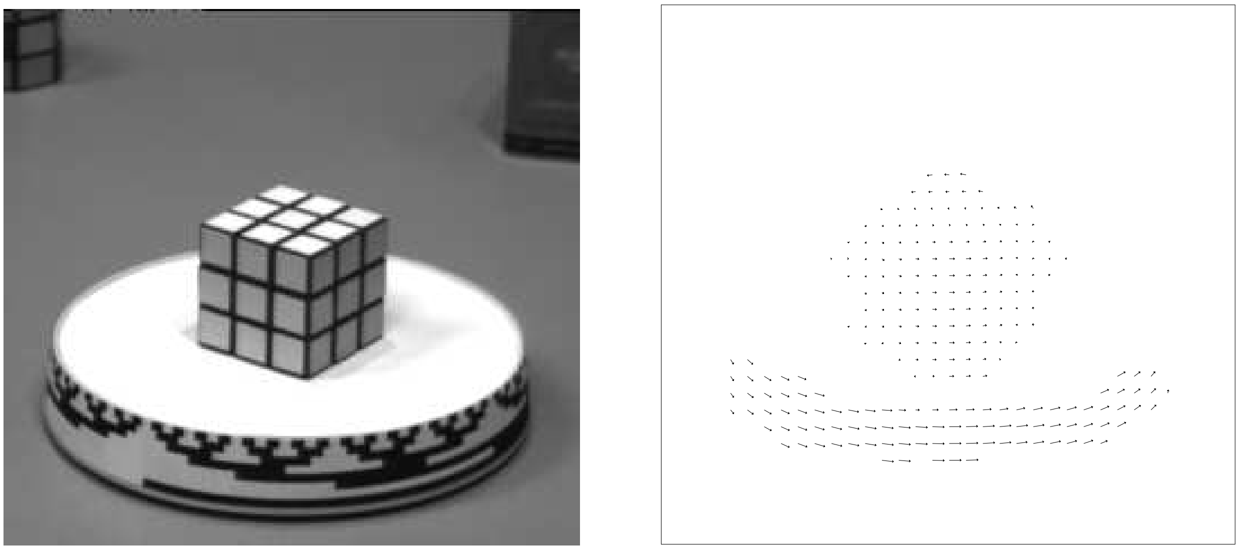


Figure 13: **Rubik Cube** sequence, sparse flow

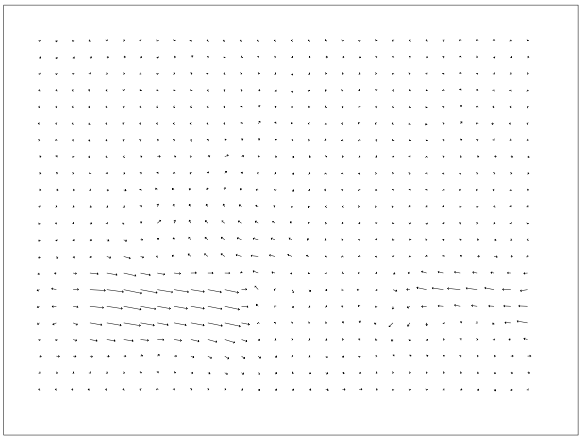
 

Figure 14: **Hamburg Taxi** sequence, dense local flow

# 9 Discussion

The spline-based motion estimation algorithms introduced in this paper are a hybrid of local optic flow algorithms and global motion estimators, utilizing the best features of both approaches. Like other local methods, we can produce detailed local flow estimates which perform well in the presence of independently moving objects and large depth variations. Unlike correlation-based methods, however, we do not assume a local translational model in each correlation window. Instead, the pixel motion within each of our patches can model affine or even more complex motions (e.g., bilinear interpolation of the four spline control vertices can provide an approximation to local projective flow). This is especially important when we analyze extended motion sequences, where local intensity patterns can deform significantly. Our technique can be viewed as a generalization of affine patch trackers [Rehg and Witkin, 1991; Shi and Tomasi, 1994] where the patch corners are stitched together over the whole image.

Another major difference between our spline-based approach and correlation-based approaches is in computational efficiency. Each pixel in our approach only contributes its error to the 4 spline control vertices influencing its displacement, whereas in correlation-based approaches, each pixel contributes to $m^2$ overlapping windows. Furthermore, operations such as inverting the local Hessian or computing the contribution to a global model only occur at the spline control vertices, thereby providing an $O(m^2)$ speedup over correlation-based techniques. For typically-sized patches ($m = 8$), this can be significant. The price we pay for this efficiency is a slight decrease in the resolution of the computed flow field, especially when compared to locally adaptive widows [Okutomi and Kanade, 1992] (which are extremely computationally demanding). However, since window-based approaches produce highly correlated estimates anyway, we do not expect this difference to be significant.

Compared to spatio-temporal filtering approaches, we see a similar improvement in computational efficiency. Separable filters can reduce the complexity of computing the required local features from $O(m^3)$ to $O(m)$, but these operations must still be performed at each pixel. Furthermore, a large number of differently tuned filters are normally used. Since the final estimates are highly correlated anyway, it just makes more computational sense to perform the calculations on a sparser grid, as we do.

Because our spline-based motion representation already has a smoothness constraint built in, regularization, which requires many iterations to propagate local constraints, is not usually necessary. If we desire longer-range smoothness constraints, regularization can easily be added to our framework. Having fewer free variables in our estimation framework leads to faster convergence when iteration is necessary to propagate such constraints.

Turning to global motion estimation, our motion model for planar surface flow can handle arbitrarily large motions and displacements, unlike the instantaneous model of [Bergen *et al.*, 1992]. We see this as an advantage in many situations, e.g., in compositing multiple views of planar surfaces [Szeliski, 1994a]. Furthermore, our approach does not require the camera to be calibrated and can handle temporally-varying internal camera parameters. While our flow field is not linear in the unknown parameters, this is not significant, since the overall problem is non-linear and requires iteration.

Our mixed global/local (rigid body) model shares similar advantages over previously developed direct methods: it does not require camera calibration and can handle time-varying camera parameters and arbitrary camera displacements. Furthermore, experimental evidence from some related structure from motion research [Szeliski, 1994b] suggests that our projective formulation of structure and motion converges more quickly than traditional Euclidean formulations.

Our experimental results suggest that our techniques are competitive in quality with the best currently available motion estimators examined in [Barron *et al.*, 1994], especially when additional regularization is used. A more complete experimental evaluation remain to be done.

# 10 Future work and Conclusions

To improve the performance of our algorithm on difficult scenes with repetitive textures, we are planning to add local search, i.e., to evaluate several possible displacements instead of just relying on gradient descent [Anandan, 1989; Singh, 1990]. We also plan to study hierarchical basis functions as an alternative to coarse-to-fine estimation [Szeliski, 1990]. This approach has proven to be very effective in other vision problems such as surface reconstruction and shape from shading where smoothness or consistency constraints need to be propagated over large distances [Szeliski, 1991]. It is unclear, however, if this is a significant problem in motion estimation, especially with richly textured scenes. Finally, we plan to address the problems of discontinuities and occlusions [Geiger *et al.*, 1992], which must be resolved for any motion analysis system to be truly useful.

In terms of applications, we are currently using our global flow estimator to register multiple 2D images, e.g., to align successive microscope slice images or to composite pieces of flat scenes such as whiteboards seen with a video camera [Szeliski, 1994a].

We plan to use our local/global model to extract 3D projective scene geometry from multiple images. We would also like to study the performance of our local motion estimator in extended motion sequences as a parallel feature tracker, i.e., by using only estimates with high local confidence. Finally, we would like to test our spline-based motion estimates as predictors for motion-compensated video coding as an alternative to block-structured predictors such as MPEG.

To summarize, spline-based image registration combines the best features of local motion models and global (parametric) motion models. The size of the spline patches and the order of spline interpolation can be used to vary smoothly between these two extremes. The resulting algorithm is more computationally efficient than correlation-based or spatio-temporal filter-based techniques while providing estimates of comparable quality. Purely global and mixed local/global estimators have also been developed based on this representation for those situations where a more specific motion model can be used.

# References

[Adelson and Bergen, 1985] E. H. Adelson and J. R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America*, A 2(2):284–299, February 1985.

[Amit, 1993] Y. Amit. A non-linear variational problem for image matching. 1993. unpublished manuscript (from Newton Institute).

[Anandan, 1989] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283–310, January 1989.

[Bajcsy and Broit, 1982] R. Bajcsy and C. Broit. Matching of deformed images. In *Sixth International Conference on Pattern Recognition (ICPR'82)*, pages 351–353, IEEE Computer Society Press, Munich, Germany, October 1982.

[Bajcsy and Kovacic, 1989] R. Bajcsy and S. Kovacic. Multiresolution elastic matching. *Computer Vision, Graphics, and Image Processing*, 46:1–21, 1989.

[Barnard and Fischler, 1982] S. T. Barnard and M. A. Fischler. Computational stereo. *Computing Surveys*, 14(4):553–572, December 1982.

[Barron *et al.*, 1994] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, January 1994.

[Beier and Neely, 1992] T. Beier and S. Neely. Feature-based image metamorphosis. *Computer Graphics (SIGGRAPH'92)*, 26(2):35–42, July 1992.

[Bergen *et al.*, 1992] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Second European Conference on Computer Vision (ECCV'92)*, pages 237–252, Springer-Verlag, Santa Margherita Liguere, Italy, May 1992.

[Blake *et al.*, 1993] A. Blake, R. Curwen, and A. Zisserman. A framework for spatio-temporal control in the tracking of visual contour. *International Journal of Computer Vision*, 11(2):127–145, October 1993.

[Bolles *et al.*, 1987] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1:7–55, 1987.

[Brown, 1992] L. G. Brown. A survey of image registration techniques. *Computing Surveys*, 24(4):325–376, December 1992.

[Burr, 1981] D. J. Burr. A dynamic model for image registration. *Computer Graphics and Image Processing*, 15(2):102–112, February 1981.

[Burt and Adelson, 1983] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, COM-31(4):532–540, April 1983.

[Carlbom *et al.*, 1991] I. Carlbom, D. Terzopoulos, and K. M. Harris. Reconstructing and visualizing models of neuronal dendrites. In N. M. Patrikalakis, editor, *Scientific Visualization of Physical Phenomena*, pages 623–638, Springer-Verlag, New York, 1991.

[Dhond and Aggarwal, 1989] U. R. Dhond and J. K. Aggarwal. Structure from stereo—a review. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1489–1510, November/December 1989.

[Dreschler and Nagel, 1982] L. Dreschler and H.-H. Nagel. Volumetric model and 3D trajectory of a moving car derived from monocular tv frame sequences of a stree scene. *Computer Graphics and Image Processing*, 20:199–228, 1982.

[Enkelmann, 1988] W. Enkelmann. Investigations of multigrid algorithms for estimation of optical flow fields in image sequences. *Computer Vision, Graphics, and Image Processing*, :150–177, 1988.

[Faugeras, 1992] O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Second European Conference on Computer Vision (ECCV'92)*, pages 563–578, Springer-Verlag, Santa Margherita Liguere, Italy, May 1992.

[Fleet and Jepson, 1990] D. Fleet and A. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5:77–104, 1990.

[Geiger *et al.*, 1992] D. Geiger, B. Ladendorf, and A. Yuille. Occlusions and binocular stereo. In *Second European Conference on Computer Vision (ECCV'92)*, pages 425–433, Springer-Verlag, Santa Margherita Liguere, Italy, May 1992.

[Gennert, 1988] M. A. Gennert. Brightness-based stereo matching. In *Second International Conference on Computer Vision (ICCV'88)*, pages 139–143, IEEE Computer Society Press, Tampa, Florida, December 1988.

[Goshtasby, 1986] A. Goshtasby. Piecewise linear mapping functions for image registration. *Pattern Recognition*, 19(6):459–466, 1986.

[Goshtasby, 1988] A. Goshtasby. Image registration by local approximation methods. *Image and Vision Computing*, 6(4):255–261, November 1988.

[Hanna, 1991] K. J. Hanna. Direct multi-resolution estimation of ego-motion and structure from motion. In *IEEE Workshop on Visual Motion*, pages 156–162, IEEE Computer Society Press, Princeton, New Jersey, October 1991.

[Hartley and Gupta, 1993] R. Hartley and R. Gupta. Computing matched-epipolar projections. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pages 549–555, IEEE Computer Society, New York, New York, June 1993.

[Hartley *et al.*, 1992] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'92)*, pages 761–764, IEEE Computer Society Press, Champaign, Illinois, June 1992.

[Heeger, 1987] D. J. Heeger. Optical flow from spatiotemporal filters. In *First International Conference on Computer Vision (ICCV'87)*, pages 181–190, IEEE Computer Society Press, London, England, June 1987.

[Hildreth, 1986] E. C. Hildreth. Computing the velocity field along contours. In N. I. Badler and J. K. Tsotsos, editors, *Motion: Representation and Perception*, pages 121–127, North-Holland, New York, New York, 1986.

[Horn and Weldon Jr., 1988] B. K. P. Horn and E. J Weldon Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, 1988.

[Horn and Schunck, 1981] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[Kass *et al.*, 1988] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, January 1988.

[Koenderink and van Doorn, 1991] J. J. Koenderink and A. J. van Doorn. Affine structure from motion. *Journal of the Optical Society of America A*, 8:377–385538, 1991.

[Le Gall, 1991] D. Le Gall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):44–58, April 1991.

[Lucas, 1984] B. D. Lucas. *Generalized Image Matching by the Method of Differences*. PhD thesis, Carnegie Mellon University, July 1984.

[Lucas and Kanade, 1981] B. D. Lucas and T. Kanade. An iterative image registration technique with an application in stereo vision. In *Seventh International Joint Conference on Artificial Intelligence (IJCAI-81)*, pages 674–679, Vancouver, 1981.

[Manmatha and Oliensis, 1992] R. Manmatha and J. Oliensis. *Measuring the affine transform — I: Scale and Rotation*. Technical Report 92-74, University of Massachussets, Amherst, Massachussets, 1992.

[Matthies *et al.*, 1989] L. H. Matthies, R. Szeliski, and T. Kanade. Kalman filter-based algorithms for estimating depth from image sequences. *International Journal of Computer Vision*, 3:209–236, 1989.

[Menet *et al.*, 1990] S. Menet, P. Saint-Marc, and G. Medioni. B-snakes: implementation and applications to stereo. In *Image Understanding Workshop*, pages 720–726, Morgan Kaufmann Publishers, Pittsburgh, Pennsylvania, September 1990.

[Mohr *et al.*, 1993] R. Mohr, L. Veillon, and L. Quan. Relative 3D reconstruction using multiple uncalibrated images. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'93)*, pages 543–548, New York, New York, June 1993.

[Nagel, 1987] H.-H. Nagel. On the estimation of optical flow: Relations between different approaches and some new results. *Artificial Intelligence*, 33:299–324, 1987.

[Okutomi and Kanade, 1992] M. Okutomi and T. Kanade. A locally adaptive window for signal matching. *International Journal of Computer Vision*, 7(2):143–162, April 1992.

[Okutomi and Kanade, 1993] M. Okutomi and T. Kanade. A multiple baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.

[Poggio *et al.*, 1985] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(6035):314–319, 26 September 1985.

[Press *et al.*, 1992] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, England, second edition, 1992.

[Quam, 1984] L. H. Quam. Hierarchical warp stereo. In *Image Understanding Workshop*, pages 149–155, Science Applications International Corporation, New Orleans, Louisiana, December 1984.

[Rehg and Witkin, 1991] J. Rehg and A. Witkin. Visual tracking with deformation models. In *IEEE International Conference on Robotics and Automation*, pages 844–850, IEEE Computer Society Press, Sacramento, California, April 1991.

[Sethi and Jain, 1987] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1):56–73, January 1987.

[Shi and Tomasi, 1994] J. Shi and C. Tomasi. Good features to track. In *IEEE Computer Society*

*Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, Washington, June 1994.

[Simoncelli *et al.*, 1991] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. Probability distributions of optic flow. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'91)*, pages 310–315, IEEE Computer Society Press, Maui, Hawaii, June 1991.

[Singh, 1990] A. Singh. An estimation-theoretic framework for image-flow computation. In *Third International Conference on Computer Vision (ICCV'90)*, pages 168–177, IEEE Computer Society Press, Osaka, Japan, December 1990.

[Szeliski, 1989] R. Szeliski. *Bayesian Modeling of Uncertainty in Low-Level Vision*. Kluwer Academic Publishers, Boston, Massachusetts, 1989.

[Szeliski, 1990] R. Szeliski. Fast surface interpolation using hierarchical basis functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):513–528, June 1990.

[Szeliski, 1991] R. Szeliski. Fast shape from shading. *CVGIP: Image Understanding*, 53(2):129–153, March 1991.

[Szeliski, 1994a] R. Szeliski. *Image Mosaicing for Tele-Reality Applications*. Technical Report 94/2, Digital Equipment Corporation, Cambridge Research Lab, June 1994.

[Szeliski, 1994b] R. Szeliski. A least squares approach to affine and projective structure and motion recovery. (in preparation) 1994.

[Szeliski and Kang, 1994] R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using nonlinear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[Terzopoulos, 1986] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4):413–424, July 1986.

[Tomasi and Kanade, 1992] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[Witkin *et al.*, 1987] A. Witkin, D. Terzopoulos, and M. Kass. Signal matching through scale space. *International Journal of Computer Vision*, 1:133–144, 1987.

[Wolberg, 1990] G. Wolberg. *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos, California, 1990.

[Xu *et al.*, 1987] G. Xu, S. Tsuji, and M. Asada. A motion stereo method based on coarse-to-fine control strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(2):332–336, March 1987.

[Zheng and Chellappa, 1992] Q. Zheng and R. Chellappa. *Automatic Feature Point Extraction and Tracking in Image Sequences for Arbitrary Camera Motion*. Technical Report CAR-TR-628, Computer Vision Laboratory, Center for Automation Research, University of Maryland, June 1992.